

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

DEPARTAMENTO DE INFORMÁTICA

UMA APLICAÇÃO DE COMPUTADOR NA EDUCAÇÃO

por

MARIA HELOISA PENEDO

Tese submetida como requisito
parcial para a obtenção do
grau de

MESTRE EM CIÊNCIAS

EM

INFORMÁTICA

Rio de Janeiro, GB, setembro de 1974

Este trabalho, realizado no Departamento de Informática da PUCRJ, contou, para a sua realização, com auxílios concedidos pelo Conselho Nacional de Pesquisas (CNPQ).

Agradecimentos

A Suely Mendes dos Santos, pela orientação segura e amiga.

A Fírmio Freire, pela ajuda na descrição do modelo de máquina.

A Sergio Eduardo Rodrigues Carvalho, pela orientação crítica.

Aos professores e funcionários, pela colaboração e estímulo.

A todos os colegas e amigos, pelo apoio e incentivo.

SUMÁRIO

O presente trabalho consiste de três partes interrelacionadas sobre a aplicação do computador na educação.

A primeira parte apresenta um resumo sobre alguns tópicos importantes em CAI (instrução assistida por computador) tais como: vantagens e desvantagens de tais sistemas, alguns exemplos de sistemas em funcionamento e projetos em desenvolvimento, e a aplicação dos mesmos em educação elementar.

Em seguida é apresentado um modelo geral de um sistema de ensino automatizado. Tal modelo contém estratégias fixas de apresentação de lições, estrutura de currículo e avaliação do desempenho do aluno.

Finalmente é descrito um modelo simplificado de computador. Ele foi projetado para ensinar às crianças, na faixa de 10 a 15 anos, certos conceitos básicos de computação. Também é fornecida a estrutura de um curso para ensinar o modelo de computador em função das estratégias mencionadas acima.

ABSTRACT

This work consists of three interrelated parts concerning Computer Assisted Instruction (CAI).

The first part provides a review of the important topics regarding CAI such as: the advantages and disadvantages of such systems, systems currently available or under development, and the application of CAI to elementary school education.

Second, a general model of computer based instruction is presented together with fixed strategies for lesson presentation, course structure, and student performance evaluation.

Lastly, a simplified computer model is developed in order to teach basic concepts concerning the digital computer to students in the ten to fifteen year old range. A course curriculum and lesson plan were developed for teaching the computer model according to the strategies outlined above.

ÍNDICE

Introdução	1
Capítulo 1. RESUMO SOBRE CAI	3
1.1. Computador na educação	3
1.2. Exemplos de projetos em curso	4
1.3. Aspectos favoráveis e dificuldades básicas	6
1.4. Aspectos relevantes de Inteligência Artificial	8
1.5. Computador Laboratório	8
1.6. Computador para o desenvolvimento do pensamento	9
1.7. Pesquisa no M.I.T.	10
1.8. Experimentação em Stanford	13
1.9. Considerações do autor	15
Capítulo 2. MODELO GERAL DE UM SISTEMA DE ENSINO AUTOMATIZADO	17
2.1. Introdução	17
2.2. Objetivos	17
2.3. Descrição das características fundamentais do sistema ...	18
2.4. Funcionamento do sistema	19
2.5. Planejamento de um Curso	21
2.6. Estruturação do Curso	22
2.6.1. Estrutura básica do curso	22
2.6.2. Fases da estruturação do curso	23
2.7. Estruturação das lições	24
2.7.1. Estrutura básica de uma lição	24
2.7.2. Estrutura básica de um tópico	25
2.7.3. Estratégia básica de apresentação	27
2.7.4. Fases da estruturação das lições	29
2.8. Avaliação	30
2.8.1. Avaliação do desempenho do aluno	30
2.8.2. Elaboração da avaliação formativa	32
2.8.3. Sugestão para a sequência de apresentação	33
2.8.4. Sugestão para critérios de avaliação	35

2.9.	Métodos de Avaliação da resposta do aluno	36
2.9.1.	Análise das respostas	36
2.9.2.	Comentários	37
2.10.	Sugestões de Implementação Geral	38
2.10.1.	Tabela da Estrutura do Currículo	38
2.10.2.	Especificação do registro de atuação do aluno ..	39
2.10.3.	Diagrama das funções do sistema	40
2.10.4.	Abordagem axiomática da elaboração do sistema ..	41
Capítulo 3.	MODELO MIC-MAC	42
3.1.	Introdução	42
3.2.	Descrição do modelo	42
3.3.	Descrição da máquina	43
3.3.1.	Características	43
3.3.2.	Registradores de Trabalho	44
3.3.3.	Instruções	44
3.3.4.	Descrição das instruções	49
3.3.5.	Representação da Informação	55
3.3.6.	Código MM	57
3.4.	Descrição da Linguagem Assembler	58
3.4.1.	Formato de instruções	58
3.4.2.	Características dos comandos	58
3.4.3.	Pseudo - Instrução	60
3.4.4.	Formato de um programa em Linguagem Assembler ...	61
3.4.5.	Especificações das instruções	61
3.4.6.	Comentários	62
3.5.	Comandos da Linguagem do Interpretador	62
Capítulo 4.	ESTUDO DE CASO	65
4.1.	Introdução ao Estudo de caso	65
4.2.	Objetivos do curso	65
4.3.	Estruturação do Currículo	66
4.3.1.	Determinação das Unidades de Ensino	66
4.3.2.	Determinação da sequência das unidades	66
4.3.3.	Descrição das Unidades de Ensino	67
4.4.	Estruturação de uma Lição	76

4.4.1. Exemplo de lição subdividida em tópicos	76
4.4.2. Exemplo da elaboração de um tópico	77
4.5. Exemplo de exercícios	87
4.6. Sugestão sobre a preparação das aulas	88,
4.7. Exemplo de Comentário-Resposta e Ajuda Automática	93
4.8. Sugestão para o material do curso	94
Apêndice A - 1. Conjunto de instruções do MIC-MAC	96
Apêndice A - 2. Exemplo de programa na Linguagem Assembler do MIC-MAC	98
Bibliografia	101

INTRODUÇÃO

A aplicação de tecnologia em educação aparece como um requisito premente para a reestruturação da educação convencional. Neste contexto, o computador é tido como uma ferramenta valiosa para uma instrução mais individualizada e especializada.

Os projetos e pesquisas em 'instrução assistida por computador' (CAI - computer assisted instruction) foram iniciados em torno de 1960 nos Estados Unidos. Dois estágios de desenvolvimento tecnológico, a saber: construção de centros de pesquisa em universidades e experimentações em escolas, estão em franca expansão. O objetivo mais amplo é o uso extenso e variado do computador para propósitos instrucionais. O capítulo 1 apresenta um resumo sobre CAI e alguns projetos em curso.

Os primeiros sistemas de CAI foram desenvolvidos com o objetivo de eliminar tarefas cansativas do professor e utilizados como ajuda instrucional. Os sistemas mais novos são mais sofisticados; neles, a instrução é baseada no computador, que gera material instrucional em função da interação com o estudante.

Um problema crucial em CAI consiste na necessidade da elaboração de um sistema com uma linguagem específica para cada curso. O que se aponta como uma solução parcial é a distribuição de lições num sistema de formato independente. Quase nada está publicado sobre tais sistemas e sobre estratégias explícitas de instrução automatizada. Por esta razão, o autor se propôs a criar um modelo geral de sistema de ensino automatizado, com uma estratégia educacional fixa de estruturação e elaboração de um curso, de tal forma que se possa aplicar a este modelo cursos diversos. Baseado em conceitos teóricos aprendidos e vivência educacional, o autor chegou a um modelo de estruturação e apresentação no qual pode ser levada em conta a atuação do estudante para guiá-lo através do curso e das lições. Isto possibilita um reforço de aprendizagem para os alunos que encontram dificuldade e, explora a facilidade de aprendizado proporcionando um aprofundamento de conceitos para os estudantes que se situam acima da média. Este modelo é apresentado no capítulo 2, complementado por sugestões para a estruturação de um currículo em função da estrutura básica e estratégias passíveis

de serem adotadas em função das estratégias fixas do modelo.

Por outro lado, inspirado pela aplicação do computador para o desenvolvimento do pensamento em educação elementar e pelas experimentações realizadas no Massachussets Institute of Technology (MIT) e na Universidade de Stanford (descritas no capítulo 1), e reconhecendo o valor do aprendizado das noções básicas de computação como modelos para processos da vida real, o autor criou um modelo hipotético de máquina, que é apresentado no capítulo 3. É um modelo simples e de fácil visualização, destinado a ser ensinado a crianças na faixa de 10 a 15 anos de idade.

Para uma amostragem de utilização prática, o autor apresenta no capítulo 4 um estudo de caso. O curso para ensinar o modelo de máquina foi estruturado em função do modelo geral de ensino, apresentado no capítulo 1.

O objetivo real do autor é dar bases para a estruturação de um curso real para o ensino do modelo hipotético de máquina.

CAPÍTULO 1

RESUMO SOBRE CAI

1.1. Computador na Educação.

Após o aparecimento do computador, sua aplicação foi totalmente voltada para o comércio e a ciência. Sua aplicação foi encarada como atividade para suprimir trabalho, sem se preocupar com os processos de pensamento envolvidos em sintetizar soluções de computador para a resolução de problemas. Somente na última década, o computador toma sua posição certa como ferramenta para desenvolver o pensamento.

Um ponto amplamente discutido nos dias de hoje, essencialmente ligado ao desenvolvimento, é o relativo à Educação. Procura-se revolucionar a educação, porém não da melhor maneira possível. O significado básico da palavra revolução indica mudança completa do sistema, porém o que se verifica é: leis mudadas, sistemas corrigidos, técnicas inseridas; ocasionando resultados infrutíferos.

Grandes horizontes estão abertos pela combinação de desenvolvimentos em dois campos: 1) desenvolvimento da tecnologia da computação - torna possível um ensino e aprendizado apropriados para as atuais necessidades individuais do ser humano; 2) grande potencial existente para o desenvolvimento da teoria do aprendizado no campo da psicologia. Não é sem razão que os psicólogos educacionais foram dos primeiros a reconhecer o valor potencial dessas pesquisas na nova área.

A escola tradicional é inapropriada: salas de aulas tradicionais, administração tradicional do tradicional sistema de educação, etc. É possível desenvolver um novo modelo de educação, através dos sistemas educacionais baseados em computação, que oferecem inúmeras vantagens, especialmente relativas a uma completa atenção individual às necessidades e desejos de cada estudante. Porém, como toda revolução, as limitações do homem o impedem de ver o que tem à mão; daí a grande oposição a utilizar o computador para este fim.

Há, em todo o mundo, uma grande demanda por educação em massa e instrução mais individualizada, ligada à preparação especializada e à motivação para os estudantes. No entanto, esta necessidade de expansão educacional, não foi nem é atendida pela falta de crescimento paralelo da produtividade dos processos educacionais, e o preço por estudante em todos os níveis cresce tanto, que causa preocupação quanto ao futuro. Sob estas circunstâncias, não é de se admirar que as instituições educacionais tentem, cada vez mais, aumentar a produtividade educacional e enriquecer o processo instrucional pela introdução de tecnologia, especialmente a baseada em computadores de alta velocidade.

Nesta última década, pois, em vários pontos do globo, projetos e pesquisas se multiplicam. A educação baseada em computadores vem obtendo pleno êxito e os resultados de experiências aplicadas revertem na reestruturação de sistemas antigos e formulação de novos.

O desenvolvimento tecnológico de computadores recai em três estratégias : ²⁵

- 1) esforços concentrados para centros de pesquisa mais sofisticados com grande variedade e profundidade de talentos em computação;
- 2) demonstrações a serem realizadas em escolas e universidades, para servir como os primeiros passos na aquisição de experiência e compreensão das possibilidades e limitações, tanto tecnológicas como econômicas, do CAI.
- 3) uso amplo e variado de computadores para propósitos instrucionais e sistemas instalados em escolas e universidades com terminais servindo como mecanismos instrucionais.

1.2. Exemplos de projetos em curso.

Daremos alguns exemplos de projetos e pesquisas em curso, com respectivos resultados.

- a) Universidade de Stanford.²⁶

Iniciou em 1963 seus programas de pesquisa e desenvolvimento em instrução assistida por computador (CAI), dirigidos por Patrick Suppes.

Programas em curso:

- programa em matemática elementar;
- programa tutorial de matemática para o segundo ano escolar,
- programa tutorial de lógica e álgebra para 5º, 6º, 7º e 8º anos;
- programa tutorial de leitura para o primeiro ano escolar,
- programa tutorial de russo ao nível universitário;
- programa de áudio usando diálogos gerados por computador e telefones particulares como terminais;
- experimentação no ensino de linguagens de programação para crianças, através do ensino de duas linguagens: LOGO e SIMPER (vide pag. 13).

Todos os programas têm obtido ampla aceitação, com grande entusiasmo da parte dos estudantes, têm registrado êxitos e proporcionado experiências valiosas. Suppes afirma que, em 1980, 15% dos estudantes americanos terão contato diário com computadores.

b) Universidade de Illinois. ¹⁵

Teve seus programas iniciados em 1959, através do laboratório de Pesquisas Educacionais baseadas em Computação (CBERL), com os objetivos de:

- a) investigar o potencial do computador no processo instrucional;
- b) projetar um sistema econômica e educacionalmente viável, incorporando as melhores abordagens para o ensino e aprendizagem.

Foram construídos três sistemas: PLATO I, II, III, para explorar as possibilidades educacionais, sem se preocupar com a parte econômica. Aplicados em todos os níveis de educação, resultaram na criação de novas estratégias de ensino, experiências com sucesso em ambientes distintos, testes positivos de atitudes de estudan-

tes, professores e autores de lições.

Um quarto sistema foi construído, PLATO IV, com o objetivo de automatizar todos os cursos introdutórios em Ciência da Computação. O sistema constitui-se em três categorias de lições:

- i) introdutórias,
- ii) descrição da sintaxe e semântica de várias linguagens de programação,
- iii) descrição da aplicação de computadores.

Os resultados da aplicação deste sistema foram altamente positivos e levaram à esperança de que seria possível, em mais dois anos, desenvolver um sistema de tal magnitude, de modo a ser usado como um completo curso automatizado, dispensando assim, a presença de um instrutor.

Vinte alunos de uma turma de medicina foram ensinados durante um semestre pelo sistema PLATO e tiveram o mesmo aproveitamento que os alunos de turmas regulares, requerendo apenas de 1/3 a 1/2 do tempo gasto pelos outros.

1.3.- Aspectos favoráveis e dificuldades básicas.

São apontados dois fatores críticos para o desenvolvimento do CAI: falta de evidência de efetividade e falta de habilidade para prover CAI duma forma mais econômica que a instrução convencional. Pesquisas isoladas tem comprovado a efetividade de tais sistemas em termos de um menor tempo para a aquisição de conhecimentos e de uma melhoria na atuação do estudante. Dado à pouca evidência de efetividade, uma pesquisa de âmbito nacional está sendo feita pela National Science Foundation nos Estados Unidos. Quanto ao fator crítico 'custo', uma pesquisa efetuada pela Boeing Computer Services²⁴ concluiu que este fator tenderá a não ser crítico devido ao aumento gradual do custo da educação convencional, à nova tecnologia de computação e ao desenvolvimento de sistemas especializados de CAI.

Entre os aspectos positivos na aplicação de tais sistemas, podemos ressaltar os seguintes:

- 1) a natureza interativa de tais sistemas absorve a atenção e leva a um total envolvimento estudantes de todos os níveis e idades.
- 2) o aproveitamento de cada aluno é individual e ele pode selecionar estratégias de ensino e métodos de apresentação que mais lhe aprouverem.
- 3) o grande número de informações a respeito do processo de ensino e aprendizado que pode ser colhido automaticamente, oferece grande ajuda para avaliação e modificação de lições e um mecanismo para medir o efeito educacional.
- 4) o material das lições pode ser modificado por um professor, de acordo com as necessidades particulares dos estudantes.

A reação contra o novo tipo de aplicação existe, porém calcada em concepções errôneas, tais como:

- educação baseada em computador como sinônimo de instrução programada;
- educação sem valor para o desenvolvimento do pensamento crítico;
- a impossibilidade de antecipação de todas as possíveis respostas, desde que a estratégia de instrução deve ser previamente programada, torna a avaliação insegura.

Tais argumentos mostram claramente a superficialidade do conhecimento de tais opositores ao CAI. Ao contrário, as experiências têm mostrado que este tipo de educação é bem mais efetivo que a educação tradicional, em muitas situações que envolvem julgamento e interpretação de problemas complexos e avaliação, pelo próprio estudante, da realidade de suas conjuncturas.

À medida que as pesquisas tomam vulto, problemas aparecem e a maior parte dos cientistas e pesquisadores volta-se para a solução dos mesmos. Por exemplo, alguns problemas que precisam ser solucionados para permitir um nível mais profundo de interação estudante-computador e o uso extenso de CAI são:

- análise detalhada e modelo de sistemas que possam operar por um custo pequeno por estudante,

- veículos mais poderosos para guardar e enviar mensagens de áudio;
- aprofundamento da interação aluno-computador em termos de perguntas abertas e respostas construídas;
- compreensão de discurso - comandar um computador oralmente.

O problema real consiste em ainda não se possuir um bom modelo, suficiente para desenvolver a interação tutor-aluno. A dificuldade central não é a limitação da máquina, mas nossa ignorância em entender explicitamente o meio de um diálogo bem sucedido. A área de computação que deverá oferecer uma ajuda relevante para a solução destas dificuldades é a área de Inteligência Artificial.

1.4. Aspectos relevantes de Inteligência Artificial.

O objetivo primordial é construir programas de computador que exibam um comportamento que poderia ser chamado de 'comportamento inteligente', quando observado em seres humanos.

Exemplos: responder perguntas não especificamente antecipadas; manter um diálogo misto contextual com um humano num subconjunto livre do Inglês; construir perguntas sobre alguns tópicos dados.

Entre as áreas importantes de pesquisa, temos:

- estruturas do conhecimento - relevante para os trabalhos sobre estruturas de informação em semântica;
- ~~comunicação~~ em linguagem natural com computadores;
- sistemas de pergunta-resposta.

1.5. Computador Laboratório.

A idéia de que jovens podem e devem aprender a programar um computador não é nova. Muitos cientistas estão interessados em mostrar o computador como um 'laboratório de matemática' para estudantes, em universidades e escolas (Papert).

O poder do laboratório consiste em que, através da linguagem, os estudantes formulam e ativam idéias e, assim fazendo, usam o laboratório

rio para seus objetivos e vontades.

1.6. Computador para desenvolvimento do pensamento.^{19,20}

Após a abordagem geral da aplicação do computador na educação, tentaremos mostrar a utilidade do computador como uma ferramenta para desenvolver o pensamento, mais especificamente para crianças de nível elementar.

O conceito de Church - "Qualquer idéia que pode ser formalizada, pode ser estudada concretamente via um programa de computação" - foi a base para uma análise mais profunda da extensão da utilidade de um computador.

A Teoria da Computação mostra que, do ponto de vista educacional, o estudo da computação é

- a formalização de idéias como seqüências de instruções;
- um método para modelar processos da vida real;
- a construção de metáforas para descrever o processamento de informações para máquinas e homens.

Segundo Papert, grande entusiasta da educação de crianças via computador, podemos agora ter uma nova visão do sistema educacional. Para este fim, a tecnologia é usada, não na forma de máquinas para crianças processarem, mas como algo que a criança aprenderá a manipular, estender e aplicar para o desenvolvimento de projetos. Desta forma ela poderá adquirir um domínio maior e mais articulado do mundo, um senso de poder do conhecimento aplicado e uma imagem confiante e realista de si mesmo como agente intelectual.

A computação torna-se uma fonte importante de idéias para revolucionar a educação. Podem-se dar às crianças poderes para inventar e levar avante projetos, possibilitando-lhes acesso ao computador com uma linguagem de programação inteligente e clara, e modelos periféricos capazes de produzir ações em linha em tempo real ('on-line real time'), como por exemplo: vídeo a cores, programas conversacionais, jogos, programas heurísticos, etc.

Um vasto universo é aberto, tornado poderoso, quando combinado com o poder conceitual das idéias teóricas associadas à computação.

As atividades ocupacionais básicas da criança envolvem aprendizado, pensamento e diversão, porém, no ensino atual, pouco ou quase nada é dito a elas sobre isto. Porque não ensiná-las a pensar, a aprender, a se divertir? Normalmente, elas constroem suas próprias teo-rias, que lhes trarão limitações posteriores. Um exemplo típico de idéia contraproducente é: 'eu consigo aprender isto, eu não consigo aprender aquilo'. Falta-lhes a compreensão de que não compreender ou não acertar não é consequência de suas habilidades intelectuais mas sim um caminho não adequado que se usa para a aprendizagem. Outro fa-to que deriva disto é a classificação de habilidades mentais, tais co-mo: 'ele é um crânio' ou 'ele é burro'.

Para isto, Papert propõe a criação de um ambiente no qual a cri-ança se envolva extensamente em experiências tais, que lhes garantam solo fértil para o surgimento de intuições e conceitos próprios para lidar com o pensamento, aprendizado, diversão, etc. Um exemplo seria escrever programas heurísticos simples, que possam jogar jogos de es-tratégias. Outro exemplo é escrever programas para ensino, escritos, desenvolvidos e testados pelas próprias crianças. ('Um ótimo meio de aprender é ensinando'). No MIT, crianças de nível escolar considera-do normal, com idade em torno de 12 anos, em apenas três meses, utili-zando a linguagem LOGO, escreviam programas para jogar jogos e, com mais algum tempo, escreviam programas para gerar sentenças aleatórias e programas conversacionais e de ensino.

Daí advêm os papéis básicos do computador na educação elementar:

- criação de tecnologia matemática para crianças - o compu-tador neste contexto é usado, não apenas para manipular símbolos, mas para controlar processos físicos reais: mo-tores que giram, caixas que emitem sons, etc.
- criação de um modelo para 'aprendizado sobre o aprendi-zado' e outras habilidades cognitivas.
- uso da teoria da computação como fonte para pesquisa em educação elementar.

O MIT já realiza projetos e experimentações marcantes com crianças em nível elementar. Nos projetos, o objetivo central é desenvolver material suficiente na forma de tecnologia e idéias, para constituir a maior parte dum currículo total elementar em Matemática, Ciências, Música, Artes e Linguística.

Ali já foi comprovado que, trabalhando com crianças por períodos ou por um ano consecutivo, seu desenvolvimento em relação ao pensamento criativo aumenta progressivamente após tal período.

Numa síntese geral, os projetos de pesquisa englobam os seguintes objetivos:

1) Compilar exemplos do uso da tecnologia para inovação da educação.

- dar às crianças um senso de poder das idéias em geral e da ciência em particular, especialmente a Matemática (que vem sendo vítima de concepções errôneas sobre didática da Matemática). Para isto, criar modelos simples, mas que funcionem em toda a extensão, e miniteorias que ajudem a criança a se situar em relação às coisas e explorar a rede complexa de idéias sobre sistemas naturais e reais.

2) Novos conceitos de conhecimento.

- para prover material com o qual a criança possa construir. Um exemplo típico é a 'Geometria da Tartaruga' (Turtle Geometry), que provê um quadro conceitual para manipulação de objetos geométricos sem a álgebra necessária para isto no estudo tradicional.

- tentar definir outros princípios, conceitos e teoremas para chegar aos mesmos conceitos e manejo instrumental do mundo físico, por um caminho diferente do existente.

3) Uma teoria cognitiva sobre e para crianças.

- desenvolver uma teoria que trabalhe com aspectos da Inteligência Artificial, preocupando-se com o 'pensar sobre o pensamento' em geral.

- teoria com base no princípio de que as crianças devem descobrir conceitos através de processos intuitivos não verbais.

através de linguagens poderosas para descrever processos complexos. Esta idéia leva a um importante ponto metodológico sobre o tipo de experimentações em psicologia cognitiva e educação.

- exemplo: introdução do conceito de Laboratório de Aprendizagem: ensinando a entender o computador, objetiva o processo, desenvolve o conceito de descrição formal e implanta idéias que ajudam bastante nos aspectos analíticos do trabalho num laboratório de aprendizagem.

4) Estudo de heurísticas.

- treinar para a criação de pensamento heurístico - para isto, desenvolver formulações mais sistemáticas e fundamentais sobre o processo do conhecimento;

- Obs.: tal estudo é atualmente considerado como atividade de segundo plano no laboratório, porém brevemente será encarada como atividade global.

5) Relação Projetos-Escolas.

- os projetos são:

a) reformistas - produzindo novos materiais e métodos que podem ser incorporados à escola tradicional sem problemas;

b) revolucionários - produzindo uma alternativa total para a escola existente;

c) intermediários - desenvolvendo novas formas de centros de aprendizagem suplementar. Ex.: Matemática suplementar para adultos.

6) Aspectos mundanos.

- os cientistas estão convencidos de que uma produção em massa levaria o custo do computador a um nível bem econômico.
- desenvolver sistemas com entradas do tipo: terminais para crianças na idade pré-escolar; terminais para pessoas com problemas motores, etc; e saídas com sistemas de vídeo econômicos.

- Exemplo: LOGO TURTLE LAB - computador equipado com uma va

riedade de mecanismos periféricos (caixa de música, gerador de fonemas) e mecanismos mecânicos (tartaruga, motores, etc).

1.8. Experimentação em Stanford.⁴

Foi realizada na Universidade de Stanford, em 1973, por Alexander B. Cannara e Stephen A. Weyer. O objetivo da experiência foi o de expor características qualitativas do manuseio de linguagens de programação e conceitos de computação para crianças e facilidades para este fim, proporcionadas por linguagens particulares. O objetivo foi baseado na crença teórica de que, se as crianças tiverem uma compreensão das capacidades teóricas da computação, poderão usá-las para um estudo mais efetivo do seu próprio pensamento sobre o mundo.

Para alcançar este objetivo, foi usada uma estrutura tutorial, que tenta ensinar conceitos de programação, requerendo:

- laboratórios interativos (interpretadores) para a linguagem;
- currículo paralelo para ensinar conceitos;
- um meio de obter dados de cada interação de estudante;
- obtenção de dados sobre atitudes de cada estudante.

Duas linguagens - § LOGO (alto nível) e SIMPER (baixo nível) - foram desenvolvidas especificamente para ensinar crianças a programar, usando o computador PDP 10. Estas duas linguagens foram escolhidas porque são ambas fáceis de aprender, computacionalmente gerais e mutuamente dissimilares.

----- x -----

§ LOGO - Feurzeig et al (1969) - é uma linguagem baseada em procedimentos, estruturada em lista. Essencialmente, contém a linguagem BASIC popular como subconjunto, porém é superior a ela em termos de consistência matemática e clareza.

SIMPER - Lorton & Slimick (1969) - desenvolvida no Instituto de Estudos Matemáticos em Ciências Sociais (IMSSS) da Universidade de Stanford. É uma linguagem 'assembler' interpretativa para uma máquina simples decimal, cujo conjunto de instruções é típico de antigos mini-computadores.

Para uma experimentação preliminar, foi reunido um grupo de 39 voluntários, de 10 a 15 anos, para um curso de dois meses, quatro dias por semana e uma hora por dia.

A experimentação visava a fazer com que estas crianças sentissem a versatilidade e limitações de computadores; apreender como as crianças: a) pensam quando programam, b) trabalham juntas, c) reagem às novas linguagens e problemas; e verificar a influência das características duma Linguagem de Programação sobre a habilidade e motivação das crianças para aprender certos conceitos de programação e aplicá-los para solucionar problemas.

Para isto, foram dados testes iniciais, relacionando conceitos de computação a problemas reais, sendo constatadas dificuldades basicamente nos seguintes pontos:

- representação dum processo de máquina num papel como um diagrama - teste de finalização dum diagrama parcial do processo duma máquina de vender balas;
- noção de endereçamento indireto - teste para obedecer a instruções aritméticas que operavam em números num conjunto de caixas numeradas.

A partir dos testes iniciais, as crianças foram divididas em 3 turmas. A linguagem LOGO foi ensinada a uma turma, a linguagem SIMPER a outra e uma terceira turma aprendeu as duas linguagens em paralelo. Foi permitido às crianças trabalhar em grupo ou individualmente.

RESULTADOS:

Verificou-se mínima evasão, grande interesse, comprovado pelo fato de comparecerem em dias extraordinários e permanecerem além da hora prevista. Verificou-se ainda que as crianças preferem trabalhar individualmente, o que lhes dá maior desenvolvimento do poder criativo.

Durante as oito semanas de experiência, gravaram-se os programas e erros das crianças, que foram utilizados para estudo do aproveitamento e aperfeiçoamento dos interpretadores para as linguagens.

A maior dificuldade dos estudantes consistiu em entender:

- a) a avaliação de um comando por um interpretador;
- b) o controle de programa;
- c) o contexto computacional criado por um programa.

Conclui-se que o currículo deve ser acompanhado por um tutor e que mensagens inteligentes de erro evitam confusões desnecessárias e facilitam o aprendizado.

Observou-se que os estudantes que aprenderam as duas linguagens em paralelo, levaram praticamente o mesmo tempo que os outros em aprender uma só; os estudantes preferiram a linguagem LOGO, de alto nível, porém notou-se que a linguagem 'assembler' ofereceu vantagens para o entendimento de certos conceitos.

A experimentação foi essencial para estabelecer as características preliminares dum efetivo sistema tutorial e desenvolver o currículo básico e meios de avaliação.

1.9. Considerações do autor.

O autor acredita que qualquer pessoa de bom senso e abertura, ao tomar conhecimento destas novas perspectivas, pode visualizar sua importância e seus aspectos amplamente positivos.

Uma mudança radical no processo de educação enfrentará inúmeras dificuldades, algumas já existentes e muitas outras a aflorar, porém, num futuro não tão remoto, ela se tornará realidade.

Para isto, há necessidade de adeptos entusiastas à causa, pessoal qualificado e numeroso, muita pesquisa, experimentação e equilíbrio suficiente para que bases sólidas sejam estabelecidas. Muitas mudanças se frustram por não estarem devidamente construídas sobre bases sólidas ou por utilizarem meios ineficazes de aplicação.

A aplicação na educação elementar, se preencher os objetivos até então definidos, poderia ser considerada como a 'ideal'. A criança poderá encontrar dificuldades no início, seguida, porém, de rápido crescimento e desenvolvimento de habilidades cognitivas, especialmente por se lhe proporcionar ambiente diferente do ambiente competitivo da sala de aula.

Atualmente, verifica-se uma obsessão pela formalização, em vez

da aplicação da intuição e heurística sobre o assunto. O método ideal de ensino deve ser ligar o conceito a ser introduzido a conceitos distintos que envolvam a mesma idéia. A 'Turtle Geometry' nos dá uma boa idéia de como isto pode ser levado a efeito. Assim vemos que o computador pode agir como o meio poderoso de desenvolvimento de 'pensamento sobre o pensamento', através dum aprendizado ligado intimamente a experiências de vida.

CAPÍTULO 2

MODELO GERAL DE UM SISTEMA DE ENSINO AUTOMATIZADO

2.1. Introdução.

Neste capítulo é apresentado um modelo geral de sistema de ensino automatizado, passível à aplicação a cursos variados, visando a preencher as necessidades individuais de cada estudante.

Ele será apresentado a seguir, complementado por estratégias de ensino adaptáveis ao modelo e sugestões para o planejamento e implementação do mesmo.

2.2. Objetivos.

Objetivo Geral: Criar um modelo geral para um sistema de ensino automatizado.

Objetivos operacionais:

- para aplicação: O modelo deve ser viável econômica e funcionalmente.
- para projetista: O sistema deve prover meios para
 - avaliação final do mesmo,
 - reestruturação.
- para professor: O sistema deve
 - fornecer relatórios sobre o progresso dos estudantes através do curso;
 - permitir reestruturação do curso.
- para aluno: O sistema deve
 - guiar o estudante através dos conceitos do curso, num andamento suficientemente rápido, para que ele não seja

- desmotivado, e suficientemente devagar, para que ele não se confunda;
- prover, para que a instrução seja altamente individualizada;
- monitorizar as respostas do estudante;
- tornar o aprendizado efetivo:
 - a) mais ativo que passivo;
 - b) fornecer 'feedback' imediato.
- ser dimensionado de tal forma que nenhum conceito importante deixe de ser ensinado de maneira adequada.

2.3. Descrição das características fundamentais do sistema.

1) Capacidade de fornecer registro de atuação do aluno a qualquer momento.

O professor poderá requerer o registro de atuação de um aluno ou de toda a turma, efetuado pelo sistema, possibilitando uma avaliação formativa que apontará as dificuldades inerentes à turma ou a alunos em particular.

2) Flexibilidade do sistema para controle do progresso do aluno.

O sistema manterá um registro detalhado da atuação do aluno em cada conceito introduzido. O progresso do aluno no curso será função de sua atuação passada e presente. Isto possibilitará fornecer ao aluno um material baseado nas suas necessidades individuais.

3) Uso da atuação passada do aluno para guiá-lo através das lições e do curso.

O estudante poderá receber todo o reforço de que necessite para os conceitos em que encontrar mais dificuldade, e mover-se rapidamente através de conceitos que já conheça ou tenha aprendido com facilidade.

4) Capacidade de suprir problemas para cada aluno, resolvê-los e mo-

nitorizar a solução de tais problemas.

O sistema fornece 'feedback' imediato, o que permite ao estudante saber se acertou ou errou logo após a sua resposta; ele recebe esta informação enquanto está interessado no problema. Em caso de erro, uma ajuda pode ser dada ao aluno, proporcionando-lhe pistas para a solução. Em função de suas respostas, ele poderá ser levado a resolver problemas ou responder perguntas similares.

- 5) Capacidade de fornecer comentários inteligentes, com grau de detalhes inverso à progressão do aluno.

As respostas a problemas serão sempre comentadas, visando a um reforço de conteúdo e motivação individual; bem detalhadas no início e menos detalhadas, à medida que o aluno progride no curso.

- 6) Capacidade de gerar exercícios em grau ascendente de dificuldade numa base individual.

Os exercícios serão apresentados aos alunos, a partir dos mais simples e tornando-se gradualmente mais difíceis, de acordo com a atuação do aluno. Desta forma, o conjunto de exercícios apresentado a alunos distintos poderá ser diferente.

- 7) Determinação do tempo gasto por cada aluno em um conceito.

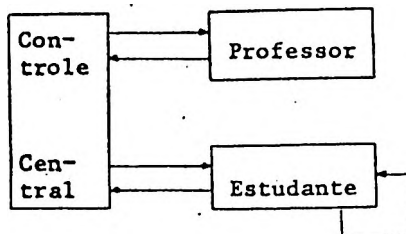
O sistema determinará e registrará o tempo gasto por cada aluno em um conceito, permitindo que se avalie a dificuldade dos mesmos, dando margem a possíveis reestruturações ou subdivisões de conceitos.

2.4. Funcionamento do sistema.

O sistema deverá operar em função de três estados: (fig. 2.1.)

- 1) Professor.
- 2) Controle Central.
- 3) Estudante.

Fig. 2.1. Estados do sistema

1) Professor.

- elabora o material instrucional:
 - a) fornece currículo do curso;
 - b) fornece lições;
 - c) fornece exercícios, problemas, respostas e comentários;
 - d) fornece critérios de avaliação.
- requer relatórios sobre a atuação de um estudante ou da turma.
- avalia lições e o curso em geral.
- reformula o curso.

2) Controle Central.

- apresenta as lições, em função do currículo.
- decide quais os conceitos a serem apresentados, em função dos critérios de avaliação pré-estabelecidos e da atuação do estudante.
- registra a atuação de cada estudante.
- fornece relatórios.

3) Estudante.

- recebe o material instrucional.
 - progride no curso.
 - responde perguntas ou resolve problemas.
- (caso seja permitida a comunicação aluno → sistema)
- requer resumos.
 - solicita lições específicas.

2.5. Planejamento de um Curso.

O planejamento de um curso a ser elaborado em função de um sistema de ensino automatizado envolverá, como qualquer sistema, três componentes importantes:

- a entrada;
- o processo;
- a saída.

Tal planejamento será uma função do 'estado professor'.

Entrada.

Será constituída da formulação dos pré-requisitos, i.e., a qualificação do aluno para fazer o curso.

A formulação dos pré-requisitos deve levar em conta a identificação do problema instrucional face

- a) às necessidades da instituição,
- b) às características da população-alvo.

A partir dos pré-requisitos, um teste deve ser aplicado aos alunos, visando à necessidade de um curso introdutório ou explicações prévias.

Processo.

Será constituído de:

- a) a descrição do programa do curso, i.e., a descrição do conteúdo,
- b) a elaboração do curso - escolha de estratégias de ensino em função do modelo de ensino.

Saída.

Constituir-se-ão dos objetivos do curso, i.e., aquilo que um aluno bem sucedido é capaz de fazer no final do curso. Estes objetivos devem explicitar as atividades observáveis desempenhadas por um aluno no decorrer e ao final do curso e um critério de avaliação para tais atividades.

2.6. Estruturação do Curso.

2.6.1. Estrutura Básica do Curso.

A estrutura básica do curso deve ser:

- 1) Assunto dividido em unidades de ensino chamadas 'lições'.
- 2) Cada unidade subdividida em tópicos.

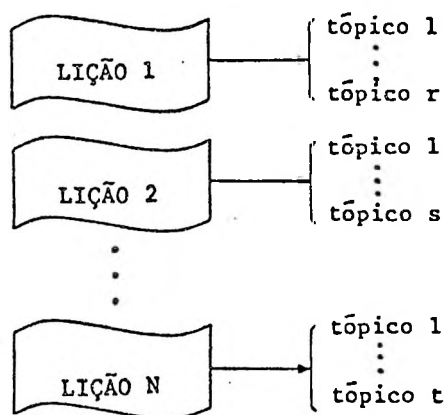


Fig. 2.2. Estrutura básica do curso.

Tendo-se tal estrutura como base, duas estratégias de ensino poderão ser adotadas, em decorrência do curso ao qual tal sistema aplicar-se-á:

Estratégia 1 - Modo pré-fixado.

A sequência das lições será pré-determinada. Neste contexto, a estrutura do currículo será linear e a ordem na qual as lições serão apresentadas a cada estudante obedecerá à sequência pré-fixada. O aluno não terá liberdade de escolha de conceitos ou lições.

Estratégia 2 - Modo escolha.

A sequência das unidades não será pré-determinada, porém certos conceitos poderão ser pré-requisitos para outros. Neste contexto, as lições devem ser agrupadas em conjuntos disjuntos (se possível) ou deve ser estruturada uma tabela de prioridades. O aluno pode, então, escolher a sequência de apresentação das lições, de acordo com tal tabela.

2.6.2. Fases da estruturação do Curso.

As fases da estruturação do curso pelo 'estado professor' podem ser definidas como:

- 1) Determinação das Unidades de Ensino.
- 2) Descrição de cada Unidade.
- 3) Determinação da sequência das unidades.
- 4) Determinação do peso de cada unidade.
- 5) Determinação de subdivisões das lições.
- 6) Determinação do tipo de problemas a serem colocados.

Descrição das Fases

1) Determinação das Unidades de Ensino.

O assunto deve ser analisado e dividido em categorias lógicas, em decorrência dos conceitos a serem apresentados e das funções a serem executadas.

Estas categorias constituirão as unidades de ensino que denominaremos 'lições'. Estas lições poderão ser agrupadas ou não, dependendo do tipo de curso e do assunto tratado.

2) Descrição de cada Unidade de Ensino.

Para cada unidade, devem ser definidos os 'objetivos'

e 'uma descrição de conteúdo', que serão de grande utilidade para a determinação da sequência das unidades e subdivisão das mesmas.

3) Determinação da sequência das unidades.

Aqui deve ser definida a estrutura do currículo. Em função da descrição de cada unidade e do objetivo geral do curso, determina-se

- a melhor sequência lógica, se Modo pré-determinado,
- os agrupamentos ou a árvore de ligação de conceitos, se Modo escolha.

4) Determinação do peso de cada unidade.

Cada unidade deve ter um peso, em função de seu conteúdo e do seu grau de complexidade, individual e na sequência.

5) Determinação da subdivisão das lições.

Em função de sua descrição, cada unidade deve ser dividida em tópicos menores, arranjados em ordem lógica, desde os mais simples até os mais difíceis.

6) Determinação do tipo de problemas a serem colocados durante a lição e a avaliação.

Em função do assunto a ser tratado e da possibilidade de avaliação pelo sistema, os tipos de exercícios devem ser determinados.

2.7. Estruturação das lições.

2.7.1. Estrutura básica de uma lição.

A estrutura básica das lições deve ser:

- 1) Lições subdivididas em tópicos.
- 2) Tópicos estruturados sob a forma de questões.

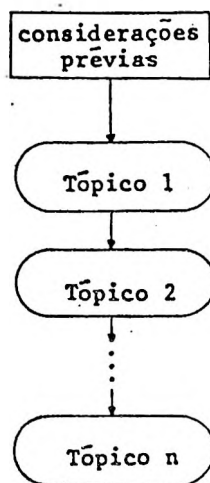


Fig. 2.3. Estrutura básica da lição.

Estratégia de ensino.

Os tópicos deverão ser escolhidos em função dos conceitos a serem apresentados na lição, e dispostos numa sequência lógica de tal forma que o tópico i seja uma decorrência imediata do tópico $i-1$. Os tópicos serão apresentados nesta sequência.

Justificativa: Esta subdivisão em tópicos subentende uma distribuição eficaz do assunto a ser ministrado, uma vez que é fato comprovado que a aprendizagem se torna mais efetiva, se distribuída em pequenas partes interligadas.¹¹

2.7.2. Estrutura básica de um tópico.

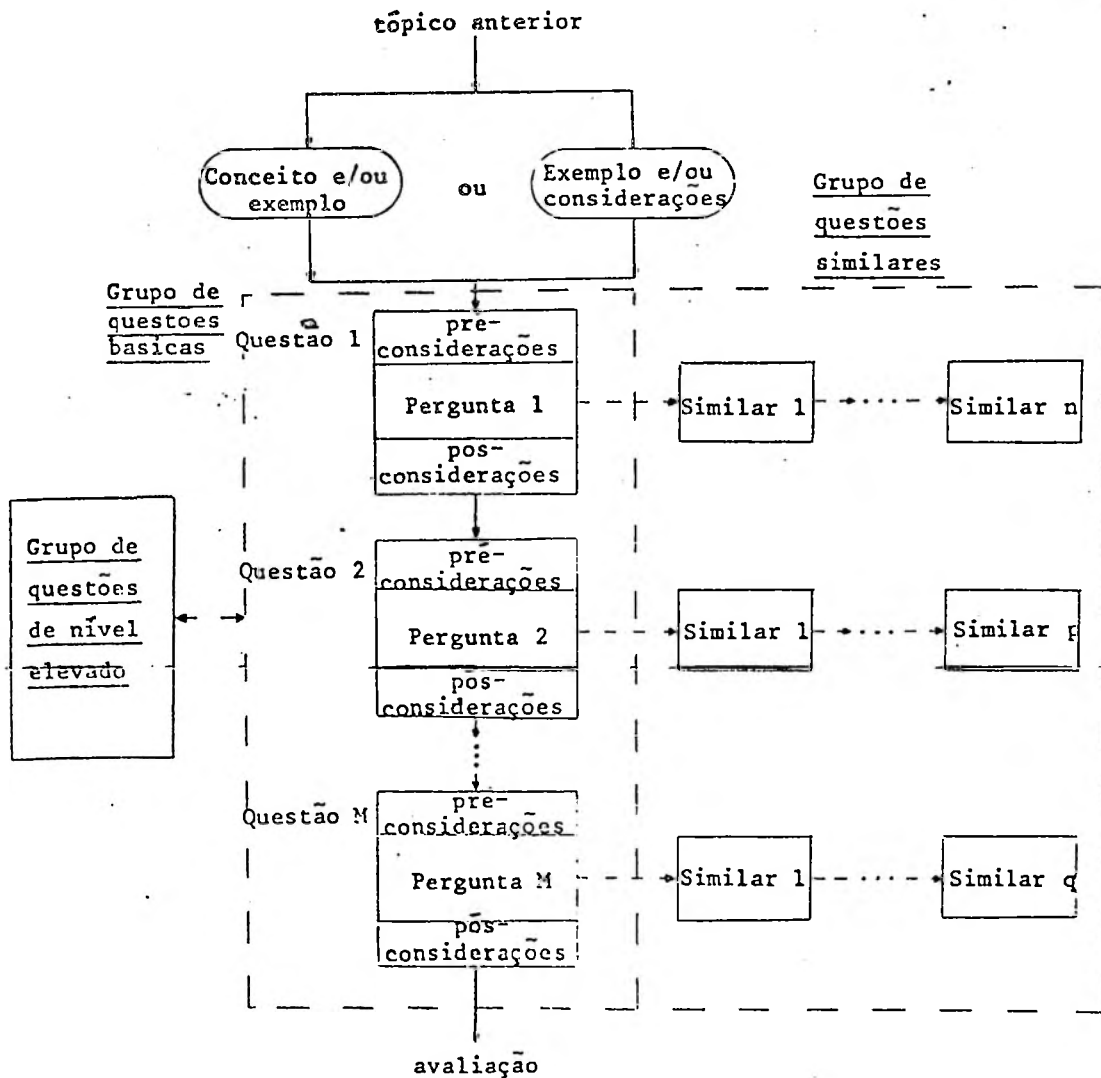
Cada tópico envolverá a apresentação de conteúdo e terá como estrutura básica grupos de questões (envolvendo o conceito

UNIVERSIDADE CATÓLICA
06/204/92 06-76

de instrução programada):

- 1) Grupo 1: a) questões básicas.
 b) questões similares.
- 2) Grupo 2: questões de nível elevado.

Fig. 2.4. Estrutura básica de um tópic.



Grupo 1:a) Questões básicas.

Estas questões envolverão o conteúdo básico mínimo a ser apresentado.

As pré e pós-considerações constituem o assunto em pauta e as perguntas, feitas em função das pré-considerações, serão consideradas como perguntas de assimilação.

b) Questões similares.

Para cada pergunta do grupo de questões básicas, devem ser elaboradas perguntas correlatas, apresentadas como reforço de assimilação, aos estudantes que não responderem corretamente às perguntas básicas.

Grupo 2:Questões de nível elevado.

Este grupo deverá ter a mesma estrutura das perguntas do Grupo 1, visando a um aprofundamento do conceito apresentado.

O objetivo é suprir o conhecimento do aluno que se situe acima da média.

2.7.3. Estratégia básica de apresentação.

Os tópicos serão apresentados na sequência pré-determinada e, em cada tópico serão apresentadas as questões pertencentes ao grupo básico.

Se todas as perguntas forem respondidas corretamente, o estudante não receberá perguntas do grupo de similares.

Em caso de erro, deve ser fornecida uma ajuda automática seguida da mesma pergunta. Se o erro persistir, o sistema fornecerá a resposta. Para cada questão, o erro (com ou sem ajuda) implicará no fornecimento da primeira pergunta similar

correlata.

A ajuda para as similares poderá ser a mesma da questão básica.

O aluno só passará para a próxima questão básica, em caso de acerto sem ajuda de pelo menos uma das similares.

Se o aluno chegar ao fim da lista de similares, isto indica que houve deficiência na aquisição do conceito. Neste caso, ele deverá retornar a algum ponto anterior, dependendo do assunto e dos critérios adotados.

Dentro do esquema da apresentação de lições (explicado a cima), duas estratégias básicas quanto à utilização do grupo de perguntas de nível elevado podem ser tomadas:

Estratégia 1 - Modo-Tempo.

Se o conjunto de objetivos se basear em:

- dar conceitos básicos;
- verificar tempo gasto para aquisição dos conceitos básicos;

então o grupo de perguntas e a avaliação de nível elevado seriam deixados para uso posterior. Ao final do curso, poderia ser dado um curso de aprofundamento aos alunos que obtivessem uma determinada média superior.

Estratégia 2 - Modo-Aprofundamento.

Se o conjunto de objetivos se basear em:

- dar o maior número de conceitos em função da capacidade e habilidade individual de cada estudante;

então o grupo de perguntas e exercícios de nível elevado seria fornecido durante o curso aos alunos que obtivessem uma determinada média.

2.7.4. Fases da estruturação das lições.

As fases da estruturação das lições podem ser definidas como:

- 1) Determinação dos tópicos e da sequência dos mesmos.
- 2) Elaboração dos tópicos.
 - 2.1) Elaboração das questões e respostas.
 - 2.2) Elaboração das similares e respostas.
 - 2.3) Elaboração da ajuda automática para cada questão e respectivas similares.
 - 2.4) Elaboração do grupo de questões de nível elevado.

Descrição das fases.

1) Determinação dos tópicos e da sequência dos mesmos.

Os tópicos devem ser determinados visando a uma subdivisão equitativa dos conceitos a serem apresentados. Seu número será função do número e da complexidade dos conceitos da lição.

Os tópicos devem ser ordenados de tal forma que constituam uma sequência lógica, partindo da introdução até o aprofundamento dos conceitos, pois eles serão apresentados nesta sequência.

2.1) Elaboração das questões e respostas.

O(s) conceito(s) pertinentes a um tópico serão apresentados sob a forma de questões, envolvendo: pré e pós-considerações e perguntas. As questões devem aparecer em grau crescente de dificuldade.

Desde que todo aluno percorrerá tais questões, o seu número deve ser função do conceito a ser introduzido e dos objetivos de tal conceito. Assim, elas devem conter o mínimo básico de ensino.

Cada tópico deve conter questões de tópicos anterior-

res correlatos, como garantia de que o estudante não falhará mais tarde.

Por 'perguntas' consideramos: perguntas objetivas, preenchimento de lacunas, etc, visando a levar o aluno a assimilar bem o conceito.

2.2) Elaboração das similares e respostas.

Para cada questão deve ser elaborado um grupo de perguntas similares. O número de tais perguntas depende do grau de dificuldade da questão à qual ela se relaciona.

O objetivo de tais perguntas é permitir o reforço do conceito em função da dificuldade do estudante.

2.3) Elaboração da ajuda automática.

Em caso de erro, o sistema deve suprir, para cada pergunta, uma ajuda na forma de exemplos e comentários, de tal modo que leve o aluno a descobrir o porquê de seu erro e a resposta correta.

2.4) Elaboração do grupo de questões de nível elevado.

A estruturação do grupo de questões de nível elevado é similar à estruturação do grupo de perguntas apresentado. Estas questões devem ser formuladas num grau de maior profundidade, envolvendo o conceito introduzido no tópico.

Este grupo só seria apresentado a alunos que obtivessem uma média alta no grupo de perguntas obrigatório, com o objetivo de aumentar o conhecimento e criar problemas mais difíceis para determinados alunos.

2.8. Avaliação.

2.8.1. Avaliação do desempenho do aluno.

A avaliação do desempenho do aluno deverá levar em conta três tipos de avaliação:

- 1) Avaliação diagnóstica - pré-teste efetuado antes do curso para se avaliar o conhecimento prévio de cada estudante sobre o assunto a ser tratado.
- 2) Avaliação formativa - após cada tópico e/ou unidade (dependendo da complexidade e extensão dos mesmos). Esta avaliação fornecerá ao aluno e ao professor meios de conhecer o seu progresso individual e em relação à turma, e indicará os tópicos ou unidades que devem ser analisados para possível reestruturação.
- 3) Avaliação somativa - ao final do curso, para verificar o comportamento terminal de cada estudante. Ela indicará se os objetivos do curso foram atingidos.

Na realidade, a avaliação do desempenho do aluno durante o curso será feita através da avaliação formativa. Esta é a razão pela qual será feita referência especial a este tipo de avaliação.

Na estruturação dos exercícios para tal avaliação, em função dos objetivos do sistema, que é levar em conta a atuação e a capacidade individual do estudante, deverão existir vários grupos de exercícios em grau ascendente de dificuldade: grupo de exercícios básicos, grupo de similares e grupo de exercícios mais difíceis.

Devem ser fixadas regras (em função do assunto do curso a ser ministrado) únicas para todo o curso ou específicas para cada lição, de tal forma que o grupo de exercícios a ser a apresentado a um aluno dependa da média obtida pela sua atuação na lição. Desta forma, o conjunto de exercícios para um aluno poderá diferir do conjunto de exercícios para outro.

2.8.2. Elaboração da Avaliação Formativa.

Para a elaboração da avaliação formativa, sugerimos o modelo abaixo:

- 1) Elaborar três grupos de exercícios pertencentes a três níveis distintos:

- a) nível 1 - básico mínimo
- b) nível 2 - médio
- c) nível 3 - elevado

O número de exercícios em um nível deve ser proporcional ao número de exercícios em outro nível.

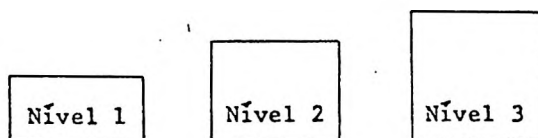


Fig. 2.3. Grupos de exercícios.

- 2) Cada grupo seria por sua vez subdividido em dois semigrupos:
 - a) o primeiro conteria todos os exercícios básicos à avaliação (dentro das características do seu nível), dispostos em ordem ascendente de dificuldade;
 - b) o segundo seria constituído de similares do primeiro (uma similar para cada exercício ba

sico), para permitir maior prática e reforço em caso de erro.

O número de exercícios para a avaliação, correspondente ao primeiro semigrupo, é função do conteúdo da lição ou tópico que está sendo avaliado.

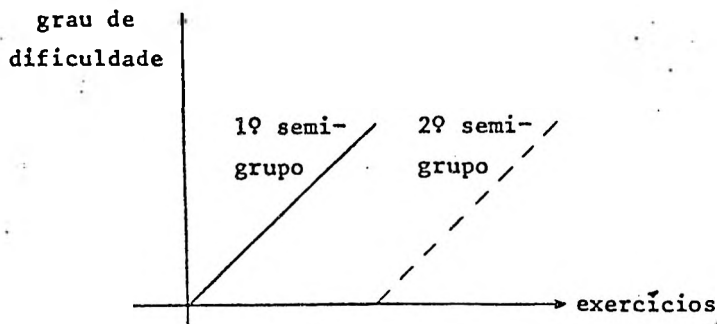


Fig. 2.6. Gradação de exercícios em um nível.

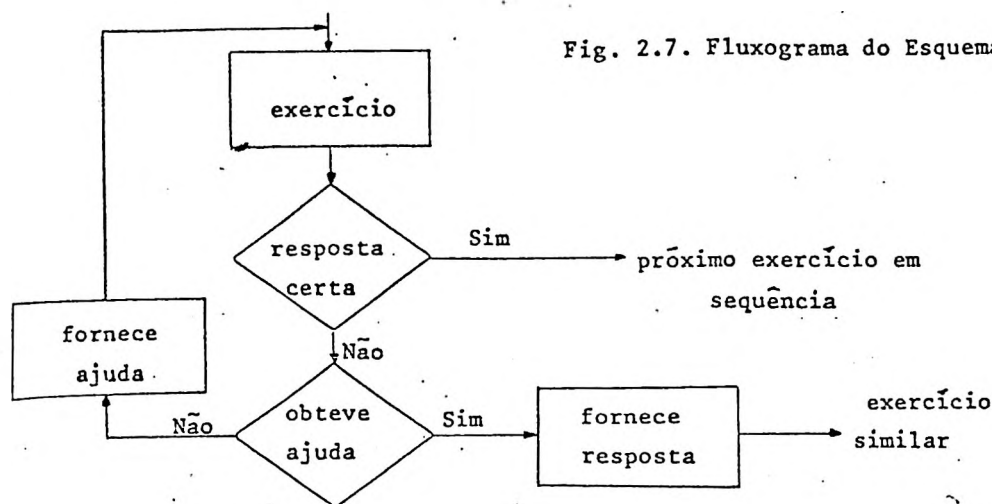
Exigência: Deve ser adotado um critério mínimo de desempenho, i.e., um número de exercícios certos (para constatar da tabela do currículo). Este número deve ser função do número de exercícios dados e o peso da lição.

2.8.3. Sugestão para a sequência de apresentação.

De acordo com o modelo apresentado, sugerimos dois esquemas de funcionamento da avaliação em um nível:

- Esquema 1:
- a) executa um exercício do primeiro semigrupo;
 - b) se acertou, passa para o próximo exercício em sequência,
se não, faz o exercício similar.

Fig. 2.7. Fluxograma do Esquema 1.



- Esquema 2: a) executa todos os exercícios do primeiro semigrupo;
 b) se obteve certa média, prossegue para a próxima lição ou tópico, se não, faz grupo de similares.

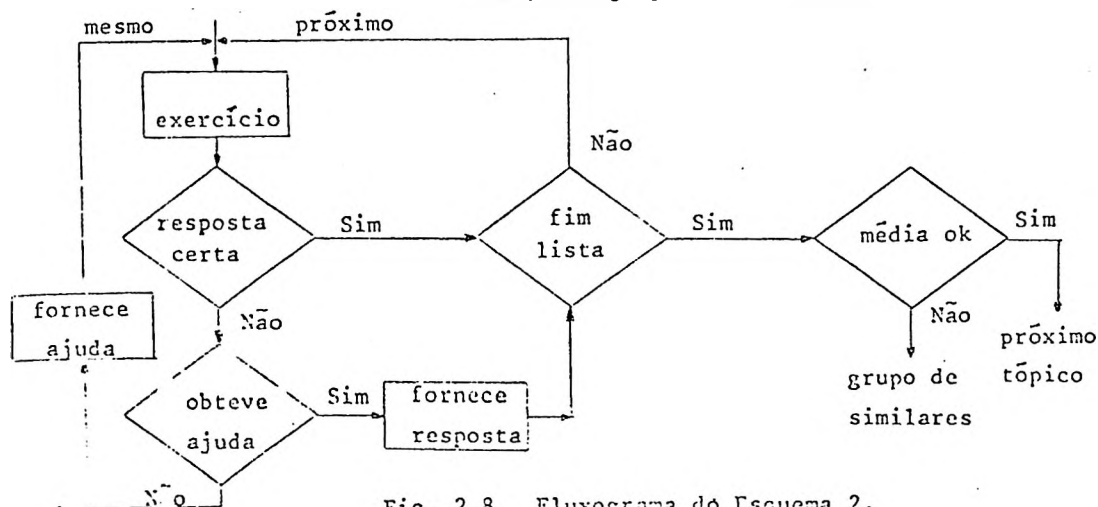


Fig. 2.8. Fluxograma do Esquema 2.

Obs.: Poderá ser oferecida uma ajuda adicional para cada exercício, em nível menos detalhado que a ajuda oferecida durante a lição.

2.8.4. Sugestão para critérios de avaliação.

Dependendo da média obtida na lição (critério a ser fixado), o aluno fará os exercícios de nível 1, nível 2 ou nível 3.

Seguem-se dois critérios sugeridos:

- 1) Para cada nível pode ser estipulado um incremento ou decremento em relação ao 'número mínimo de acertos'. Em função deste novo número, resultado do incremento ou decremento, tomam-se as seguintes medidas:

- se este número de acertos é obtido, o aluno passará para a próxima lição ou tópico e o seu desempenho é registrado,
- se este número de acertos não é alcançado,
 - a) o estudante passará para o grupo de exercícios imediatamente inferior, se em nível 2 ou 3.
 - b) o estudante deverá refazer a lição, se em nível 1.

Obs.: Deve ser fixado um número máximo de repetições para as lições (número fixo ou variável para cada lição).

- 2) Podem ser estipulados três números X, Y, Z para cada tópico ou lição, de tal forma que, ao final de Y exercícios feitos, o estudante
 - a) volta para o tópico ou lição anterior (ou grupo anterior, se existirem grupamentos de lições ou tópicos), se o seu número de acertos foi menor que X.
 - b) volta para a mesma lição, se o número de acertos foi maior ou igual a X ou menor que Z.
 - c) prossegue, se o número de acertos é maior ou i-

elaboração inteligente das perguntas colocadas.

2.9.2. Comentários.

Os comentários podem ser de dois tipos:

- a) de conteúdo ou reforço;
- b) de motivação.

Todas as respostas devem ser comentadas. No caso de respostas tipo 'Sim/Não', múltipla escolha, etc., os erros e acertos devem ser comentados em termos de conteúdo ou reforço e motivação.

No caso de respostas ambíguas, quando não se pode verificar a causa do erro, o comentário de motivação é imprescindível, e o comentário de reforço será dado pela ajuda automática.

O objetivo é elaborar comentários inteligentes, visando a levar o estudante a compreender que é capaz de solucionar problemas ou responder perguntas.

2.10. Sugestões de Implementação Geral.

2.10.1. Tabela da Estrutura do Currículo.

O currículo será constituído duma estrutura que classifica os conceitos a serem apresentados, com respectivos pesos e número mínimo de exercícios certos para cada lição.

Se em Modo Pré-fixado, a ordem de apresentação será sequencial e se em Modo Escolha, as ligações devem ser providas.

O estudante percorrerá toda a estrutura num andamento que será função de sua habilidade para perfazer corretamente cada lição.

A seleção do conceito (prosseguimento pela estrutura ou retrocesso) será individualizada em função dos critérios adotados, pelo exame do registro de atuação do aluno.

Unidade	Tópico	Peso	Número de exercícios

Fig. 2.10. Tabela de currículo.

2.10.2. Especificação do Registro de atuação do aluno.

O sistema deve manter um registro de todo o curso para cada estudante, informando a lição atual e sua atuação em relação a cada lição.

Deve prover um registro detalhado do progresso do estudante pela lição atual, para permitir que a lição ou tópico seja repetido.

Deve ser possível dar graus a cada estudante com respeito a cada lição ou tópico.

O sistema deve prover para que todos os registros sejam impressos num formato legível para facilidade do professor.

Fig. 2.11. Registro de atuação do aluno.

Nome do aluno: Número do aluno: Data última sessão: Lição atual: Número tópico atual: Média atual: Nº repetições tópico:							
Número lição	Nota	Tópico	Número repetições	Nº questões com ajuda	Nível exerc.	Nº exercs. certos	Nº exercs. com ajuda

2.10.3. Diagrama das Funções do Sistema.

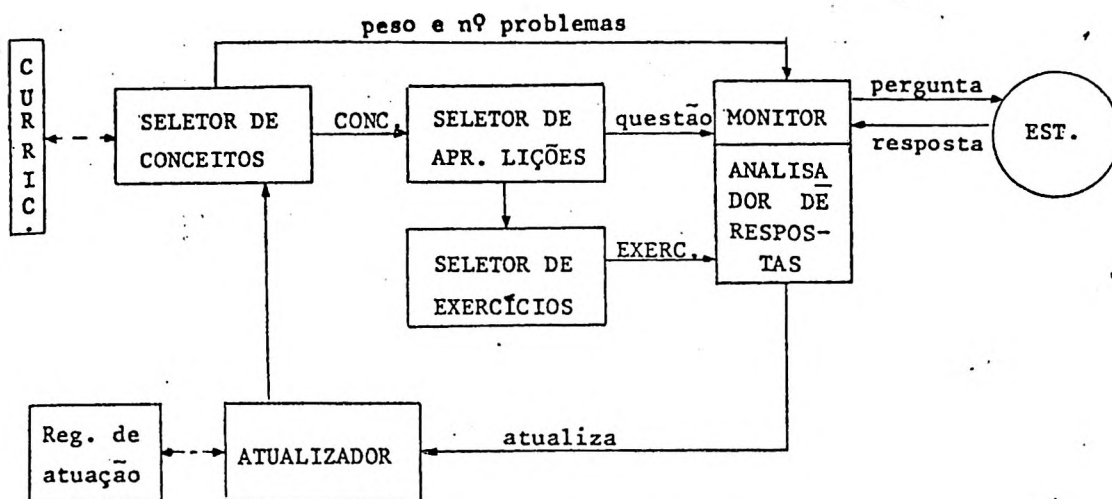


Fig. 2.12. Diagrama das funções do sistema.

Currículo - estrutura que classifica as unidades e tópicos do curso.

Atualizador - conjunto de regras para definir o progresso de cada estudante através do curso, registrando a atuação do mesmo em cada conceito introduzido.

Seletor de Apresentação de Lições - conjunto de regras para determinar a sequência de apresentação de questões.

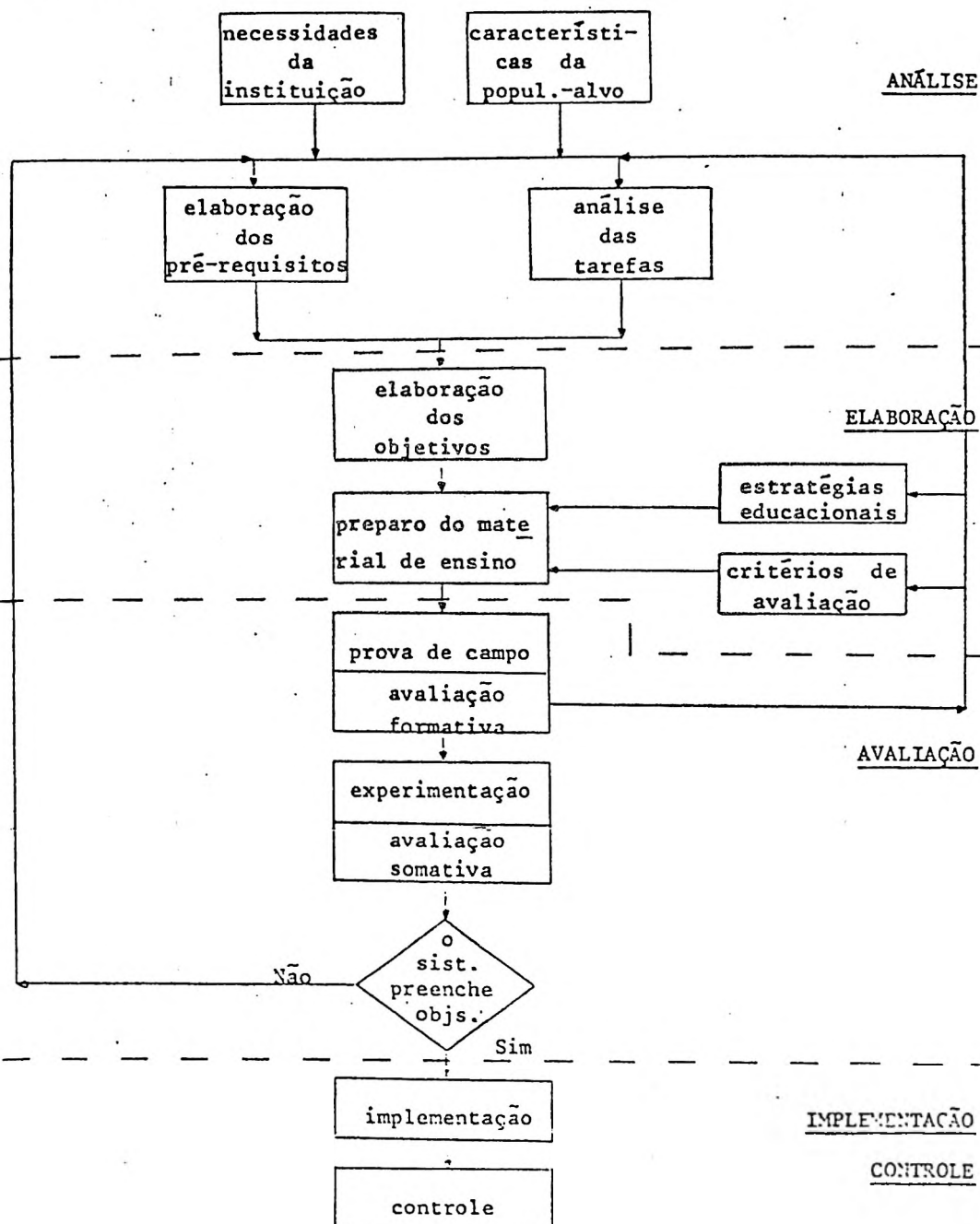
Seletor de Exercícios - conjunto de regras, em função dos critérios adotados, para determinar os exercícios a serem resolvidos para a avaliação dos conceitos.

Monitor - conjunto de regras para monitorizar a resposta de cada estudante às perguntas e estabelecer graus de acordo com o conceito.

Analizador de Respostas - conjunto de regras para, a partir das soluções fornecidas pelo 'estado professor', analisar as respostas dos estudantes e emitir respostas e/ou comentários.

2.10.4. Abordagem Axiomática da elaboração do Sistema.

Fig. 2.13.



CAPÍTULO 3

MODELO MIC-MAC

3.1. Introdução.

No Brasil são bem reduzidas as pesquisas sobre aplicação do computador em educação, especialmente no nível elementar. Por esta razão e entusiasmado com as experimentações realizadas em Stanford, o autor se propôs a criar um modelo hipotético de máquina com suas linguagens associadas que permitisse, através de seu ensino, dar às crianças, na faixa de 10 a 15 anos, certos conceitos básicos de computação, com o objetivo de gerar perspectivas para a formação de um curso real para o ensino de tais conceitos.

Várias linguagens e modelos de máquina, incluindo o modelo Simper (aplicado em Stanford) foram estudados e analisados, surgindo então um modelo hipotético ao qual denominamos MIC-MAC.

Segue-se uma descrição detalhada do MIC-MAC e suas linguagens: de máquina, 'assembler' e de controle.

3.2. Descrição do modelo.

O modelo MIC-MAC será constituído de três componentes funcionais:

- o interpretador
- o assembler
- o simulador ou computador MIC-MAC.

Uma linha fornecida por um terminal poderá conter:

- a) um comando em linguagem do interpretador;
- b) um comando em linguagem de máquina;
- c) um comando em linguagem 'assembler';
- d) dados para um programa.

Interpretador - Recebe toda linha fornecida por um terminal, analisa-a e

- executa-a, se é um comando em linguagem do interpretador;
- armazena na memória, se é um comando em linguagem de máquina;
- envia para o 'Assembler', se um comando em linguagem 'assembler';
- armazena no registrador indicado, se um dado.

Assembler - Analisa sintaticamente e traduz para linguagem de máquina os comandos da linguagem 'assembler'.

Computador MIC-MAC - Executa os programas.

3.3. Descrição da Máquina.

O computador MIC-MAC é uma máquina simples, criada com o objetivo de dar às crianças de 10 a 15 anos uma idéia funcional do comportamento de um computador.

O computador MIC-MAC é uma máquina decimal com memória capaz de armazenar 500 palavras de 8 dígitos cada. A memória é endereçada de 1 a 500. A unidade aritmética do MIC-MAC efetua quatro operações básicas: adição, subtração, multiplicação e divisão.

3.3.1. Características.

- memória pequena: 500 palavras.
- palavra de 8 dígitos.
- 4 registradores endereçáveis: A, B, C, D.
- contador de instruções endereçável: registrador P.
- 23 instruções.

3.3.2. Registradores de Trabalho.

A, B, C, D - (8 dígitos cada)

Estes registradores contêm os dados que serão usados em operações. Seu conteúdo pode ser movido ou rodado para a direita ou esquerda; incrementado ou negado; guardado na memória ou operado com conteúdos de memória; ou comparado aritmeticamente com outro registrador.

Eles guardam o resultado das operações aritméticas feitas pelas instruções.

A transferência de dados para o meio externo é realizada através destes registradores.

P - (3 dígitos)

Este registrador contém o endereço da próxima instrução a ser executada. É também um registrador endereçável.

Após a execução de uma instrução, o registrador P automaticamente é incrementado de 1, a exceção das instruções de comparação (COMP, TESTZ), a instrução de desvio incondicional (VA) e a instrução FAÇA envolvendo o registrador P.

3.3.3. Instruções.

O conjunto de instruções é constituído de 23 instruções de 8 dígitos cada, subdividido em três (3) subconjuntos (de acordo com o formato da palavra).

- a) FORMATO 1 - Este subconjunto contém instruções com referência à memória. É constituído de 4 grupos de instruções.

Formato básico:

x	x	x	x	x	x	x	x
1	2	3	4	5	6	7	8

posições 1 e 2 - código de operação.

posição 3 - nº representativo do registrador utilizado (1-A, 2-B, 3-C, 4-D, 5-P).

posição 4 - endereçamento indireto (posição 4 = 1).

posições 5 a 8 - endereço de memória.

1º GRUPO - todos os campos contêm informação.

<u>Instrução</u>	<u>Mnemônico</u>	<u>Código</u>	<u>Ação</u> *
Carrega registrador	CAR	10	$C(R) \leftarrow C(m)$
Carrega palavra de memória	LEVE	11	$C(m) \leftarrow C(R)$
Soma conteúdos de reg. e palavra de memória	SOME	15	$C(R) \leftarrow C(R) + C(m)$
Subtrai do reg. o conteúdo da palavra de mem.	SUB	16	$C(R) \leftarrow C(R) - C(m)$
Multiplica	MULT	17	$C(R) \leftarrow C(R) * C(m)$
Divide	DIV	18	$C(R) \leftarrow C(R) / C(m)$

----- x -----
 * A letra 'R' indica registrador.

A letra 'm' indica endereço de memória.

A letra 'N' indica dado numérico externo.

A letra 'C' indica dado alfanumérico externo.

'C(x)' indica conteúdo de x.

O sinal ' $\leftarrow$ ' indica atribuição.

29 GRUPO - campo de registrador nulo (posição 3 = 0).

formato:

x	x	0	x	x	x	x	x
1	2	3	4	5	6	7	8

<u>Instrução</u>	<u>Mnemônico</u>	<u>Código</u>	<u>Ação</u>
Desvio incondicional	VA	20	$C(P) \leftarrow m$

39 GRUPO - campo de endereço nulo (posições 5 a 8 = 0) e campo de endereçamento indireto nulo (pos. 4=0)

formato:

1	2	3	4	5	6	7	8
x	x	x	0	0	0	0	0

<u>Instrução</u>	<u>Mnemônico</u>	<u>Código</u>	<u>Ação</u>
Incrementa reg.	CSEC	21	$C(R) \leftarrow C(R) + 1$
Nega registrador	NEG	22	$C(R) \leftarrow - C(R)$
Se $C(R)=0$, pula próxima instrução	TESTZ	23	se $C(R)=0$, então $C(P) \leftarrow C(P) + 2$
Solicita dado numérico	NPED	25	$C(R) \leftarrow N$
Solicita caracter	CPED	26	$C(R) \leftarrow C$
Imprime número	NIMP	27	$N \leftarrow C(R)$
Imprime caracter	CIMP	28	$C \leftarrow C(R)$

49 GRUPO - campo de registrador nulo (posição 3 = 0), campo de endereçamento indireto nulo (pos. 4=0) e campo de endereço nulo (posições 5 a 8 = 0).

formato:

1	2	3	4	5	6	7	8
x	x	0	0	0	0	0	0

Instrução	Mnemônico	Código
Fim de execução	PARE	01
Nenhuma operação	NADA	00

- b) FORMATO 2 - Este subconjunto contém instruções com referência a registradores e valores.

Formato básico:

1	2	3	4	5	6	7	8
x	x	x	x	x	x	x	x

posições 1 e 2 - código de operação.

posição 3 - nº representativo de registrador.

posições 4 a 8 - valor (sem sinal).

<u>Instrução</u>	<u>Mnemônico</u>	<u>Código</u>	<u>Ação</u>
Coloca valor no registrador	FAÇA	30	$C(R) \leftarrow \text{valor}$
Move nº dígitos = valor, para esquerda	MOVE	35	----
Move nº dígitos = valor, para direita	MOVD	36	----
Roda nº dígitos = valor, para esquerda	RODE	37	----
Roda nº dígitos = valor, para direita	RODD	38	----

- c) FORMATO 3 - Este subconjunto contém 2 instruções com referência a dois registradores. As posições 4 a 7 são nulas.

Formato básico:

1	2	3	4	5	6	7	8
x	x	x	0	0	0	0	x

posições 1 e 2 - código de operação.

posição 3 - nº representativo do reg. 1.

posição 8 - nº representativo do reg. 2.

<u>Instrução</u>	<u>Mnemônico</u>	<u>Código</u>	<u>Ação</u>
Compara regs. e pula próxima instrução, se diferentes	COMP	40	se $C(R1) \neq C(R2)$, $C(P) \leftarrow C(P)+2$ se $C(R1) = C(R2)$, $C(P) \leftarrow C(P)+1$
Troca conteúdo de regs.	TROC	41	$C(R1) \leftarrow C(R2)$ e $C(R2) \leftarrow C(R1)$

3.3.4. Descrição das Instruções.

As instruções com referência à memória executam alguma operação envolvendo posições de memória; transferindo informação para ou das células de memória, efetuando operações aritméticas, etc.

Observemos que todas as posições de memória podem ser acessadas diretamente, desde que existem 4 dígitos no campo de endereço e a maior posição endereçável é 500. No entanto é possível a utilização de endereçamento indireto, com a posição 4 igual a 1.

<u>Código</u>	<u>Mnemônico</u>
20	VA
10	CAR
11	LEVE
15	SOME
16	SUB
17	MULT
18	DIV

VA - Esta instrução posiciona o registrador P de acordo com o campo de endereço da instrução.

Exemplo: VA 50

instrução de máquina:

2	0	0	0	0	0	5	0
---	---	---	---	---	---	---	---

ação: C(P) ← 50

CAR - O registrador indicado no campo de registrador é carregado com o conteúdo da posição endereçada. O conteúdo da memória permanece inalterado.

Exemplo: CAR A 10

instrução de máquina:

1	0	1	0	0	0	1	0
---	---	---	---	---	---	---	---

ação: C(A) ← C(10)

LEVE - O conteúdo do registrador indicado no campo de registrador é armazenado na posição endereçada. Perde-se o conteúdo anterior da memória; o registrador permanece inalterado. É a única instrução em que a ação não resulta na modificação do primeiro operando.

Ex.: LEVE D 135

instrução de máquina:

1	1	4	0	0	1	3	5
---	---	---	---	---	---	---	---

ação: $C(135) \leftarrow C(D)$

SOME (SUB, MULT, DIV) - soma ao (subtrai do, multiplica ao, divide do) registrador indicado no campo de registrador, o conteúdo da memória endereçada e o resultado permanece no registrador. O conteúdo da memória fica inalterado.

Ex.: SOME B 20

instrução de máquina:

1	5	2	0	0	0	2	0
---	---	---	---	---	---	---	---

ação: $C(B) \leftarrow C(B) + C(20)$

As operações com referência exclusiva a um registrador executam operações tais como incrementar, testar ou negar o conteúdo do mesmo, ou operações de entrada e saída.

<u>Código</u>	<u>Mnemônico</u>
21	CSEC
22	NEG
23	TESTZ
25	NPED
26	CPED
27	NIMP
28	CIMP

CSEC - Incrementa de 1 o conteúdo do registrador indicado no campo de registrador.

Ex.: CSEC P

instrução de máquina:

2	1	5	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: $C(P) \leftarrow C(P) + 1$

NEG - Nega o conteúdo do registrador indicado no campo de registrador. Desde que os dados numéricos são representados na forma decimal, esta instrução apenas muda o dígito 1 (primeiro) do registrador de 0 para 1 (ou vice-versa).

Ex.: NEG C

instrução de máquina:

2	2	3	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: $C(C) \leftarrow -C(C)$

TESTZ- Verifica o conteúdo do registrador. Se igual a 0, a próxima instrução é pulada, i.e., o registrador P é incrementado de 2. Se não o registrador P é incrementado de 1.

Ex.: TESTZ A

instrução de máquina:

2	3	1	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: se $C(A) = 0$ então $C(P) \leftarrow C(P) + 2$

se $C(A) \neq 0$ então $C(P) \leftarrow C(P) + 1$

NPED - Requer um dado numérico a ser colocado como conteúdo do registrador indicado no campo de registrador.

Ex.: NPED B

instrução de máquina:

2	5	2	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: supondo $N = 538\ 932$

reg. B =

0	0	5	3	8	9	3	2
---	---	---	---	---	---	---	---

 depois

CPED - Requer um dado alfanumérico, cujo código interno será colocado como conteúdo do registrador indicado no campo de registrador.

Ex.: CPED D

instrução de máquina:

2	6	4	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: supondo C = 'B'

reg. D =

0	0	0	0	0	0	6	1
---	---	---	---	---	---	---	---

 depois

NIMP - Imprime um dado numérico, conteúdo do registrador indicado no campo de registrador.

Ex.: NIMP A

instrução de máquina:

2	7	1	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: supondo

reg. A =

1	0	0	4	3	2	9	1
---	---	---	---	---	---	---	---

então N = - 43 291 depois

CIMP - Imprime um dado alfanumérico, correspondente ao código interno, que é conteúdo do registrador indicado no campo de registrador.

Ex.: CIMP B

instrução de máquina:

2	8	2	0	0	0	0	0
---	---	---	---	---	---	---	---

ação: supondo

reg. B =

4	1	0	0	1	3	4	5
---	---	---	---	---	---	---	---

então C = '?' depois

Instruções com campos de registrador e endereço de memória nulos:

PARE - (código 01) - termina a execução do programa.

NADA - (código 00) - instrução que não faz operação alguma.

As instruções que envolvem um registrador e um valor, executam alguma operação envolvendo colocação de valores em re

gistradores e operações de movimentação de dígitos. As instruções de movimento são instruções lógicas, i.e., não é preservado o dígito de sinal.

<u>Código</u>	<u>Mnemônico</u>
30	FAÇA
35	MOVE
36	MOVD
37	RODE
38	RODD

FAÇA - Coloca o valor do campo de valor como conteúdo do registrador indicado no campo de registrador. Este valor não pode ultrapassar 5 dígitos.

Ex.: FAÇA A 49

instrução de máquina:

3	0	1	0	0	0	4	9
---	---	---	---	---	---	---	---

ação: C(A) ← 49

MOVE - Move os dígitos do conteúdo do registrador indicado no campo de registrador para a esquerda, tantas vezes quantas indique o valor no campo de valor. Os dígitos de mais alta ordem que são movidos, são perdidos. Os dígitos de mais baixa ordem são preenchidos com zeros.

Ex.: MOVE A 5

instrução de máquina:

3	5	1	0	0	0	0	5
---	---	---	---	---	---	---	---

ação:

0	0	4	3	2	9	1	2
---	---	---	---	---	---	---	---

antes

reg. A

9	1	2	0	0	0	0	0
---	---	---	---	---	---	---	---

depois

MOVD - Move os dígitos do conteúdo do registrador para a direita, tantas vezes quantas indique o valor no campo de valor. Os dígitos de mais baixa ordem são perdidos.

Os dígitos de mais alta ordem são preenchidos com zeros.

Ex.: MOVD A 5

instrução de máquina:

3	6	1	0	0	0	0	5
---	---	---	---	---	---	---	---

ação:

0	0	4	3	2	9	1	2
---	---	---	---	---	---	---	---

 antes

reg. A

0	0	0	0	0	0	0	4
---	---	---	---	---	---	---	---

 depois

RODE - Instrução idêntica à instrução MOVE, apenas os dígitos de mais alta ordem não são perdidos e reaparecem à direita.

Ex.: RODE A 5

instrução de máquina:

3	7	1	0	0	0	0	5
---	---	---	---	---	---	---	---

ação:

0	0	4	3	2	9	1	2
---	---	---	---	---	---	---	---

 antes

reg. A

9	1	2	0	0	4	3	2
---	---	---	---	---	---	---	---

 depois

RODD - Instrução idêntica à instrução MOVD, apenas os dígitos de mais baixa ordem não são perdidos e reaparecem à esquerda.

Ex.: RODD A 5

instrução de máquina:

3	8	1	0	0	0	0	5
---	---	---	---	---	---	---	---

ação:

0	0	4	3	2	9	1	2
---	---	---	---	---	---	---	---

 antes

reg. A

3	2	9	1	2	0	0	4
---	---	---	---	---	---	---	---

 depois

As instruções que envolvem dois registradores, executam alguma operação envolvendo troca ou comparação dos conteúdos dos registradores.

<u>Código</u>	<u>Mnemônico</u>
40	COMP
41	TROC

COMP - Compara os conteúdos dos registradores indicados nos campos de registrador. Se as palavras são diferentes, a próxima instrução é pulada, i.e., o registrador P é incrementado de 2. Se não, o programa prossegue normalmente para a próxima instrução em sequência. O conteúdo dos registradores não é alterado.

Ex.: COMP C D

instrução de máquina:

4	0	3	0	0	0	0	4
---	---	---	---	---	---	---	---

ação: se $C(C) \neq C(D)$ então $C(P) \leftarrow C(P) + 2$

se $C(C) = C(D)$ então $C(P) \leftarrow C(P) + 1$

TROC - Troca os conteúdos dos registradores indicados nos campos de registrador.

Ex.: TROC A B

instrução de máquina:

4	1	1	0	0	0	0	2
---	---	---	---	---	---	---	---

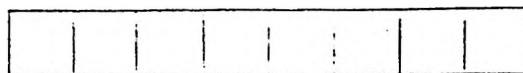
ação: $C(A) \leftarrow C(B)$ e

$C(B) \leftarrow C(A)$

3.3.5. Representação da Informação.

Números - Os números são representados internamente na sua forma decimal usual.

Formato básico:



senal

magnitude

Os dados numéricos sã podem ser números inteiros de 7 dígitos e são representados a partir da segunda posição da palavra.

Um número é positivo, se o dígito de sinal (1ª posição da palavra) for 0, e negativo, se for 1.

O maior número positivo ou negativo em valor absoluto é 9.999.999 .

Exemplos:

+ 4281 ≡

0	0	0	0	4	2	8	1
---	---	---	---	---	---	---	---

- 3892 ≡

1	0	0	0	3	8	9	2
---	---	---	---	---	---	---	---

Caracteres - Os caracteres são representados internamente pelo código MM correspondente (vide item 3.3.6.), colocado nas posições de mais baixa ordem da palavra.

Cada caracter corresponde a dois algarismos.

Uma palavra de memória pode conter no máximo 4 caracteres.

Exemplos:

car. 'A' ≡

0	0	0	0	0	0	6	0
---	---	---	---	---	---	---	---

car. '3' ≡

0	0	0	0	0	0	5	3
---	---	---	---	---	---	---	---

car. '=' ≡

0	0	0	0	0	0	0	8
---	---	---	---	---	---	---	---

3.3.6. CÓDIGO MM

<u>Caracter</u>	<u>Código</u>
.	02
,	03
;	04
:	05
—	06
<	07
=	08
>	09
?	45
@	46
¥	47
0	50
1	51
2	52
3	53
4	54
5	55
6	56
7	57
8	58
9	59
A	60
B	61
C	62
D	63
E	64
F	65
G	66
H	67
I	68

<u>Caracter</u>	<u>Código</u>
J	69
K	70
L	71
M	72
N	73
O	74
P	75
Q	76
R	77
S	78
T	79
U	80
V	81
W	82
X	83
Y	84
Z	85
'	86
"	87
# (*)	88
\$	89
%	90
&	91
	92
(	93
)	94
*	95
+	96
-	97
/	98
~	99

(*) O símbolo = indica 'mudança de linha'.

3.4. Descrição da Linguagem Assembler .

3.4.1. Formato de Instruções.

Um comando em linguagem fonte consiste de:

- a) código de operação - mnemônico de operações de máquina ou um pseudo-código de 'assembler'.
- b) um ou dois operandos - (quando requerido pela instrução) - pode ser um alfanumérico representando um registrador (A, B, C, D ou P) e/ou um termo numérico ou simbólico ou expressões.

3.4.2. Características dos Comandos.

Os campos aparecem na seguinte ordem: código de operação, seguido dos operandos.

Um ou mais espaços em branco separam os campos de um comando, porém o comprimento de um comando não deve exceder de 50 caracteres, incluindo o espaço em branco.

Um comando de fim termina o programa.

O conjunto de caracteres que podem aparecer num comando consiste de letras de A a Z e dígitos de 0 a 9, constituindo os símbolos alfanuméricos; os sinais +, -, *, / e '.' são usados em expressões aritméticas. O símbolo @ representa endereçamento indireto.

a) Código de Operação.

O código de operação define uma operação a ser feita pelo computador ou 'Assembler'. Termina por um espaço em branco.

co.

Categorias:

- | | |
|--------------------------------|-------|
| 1) Operações aritméticas: | SOME |
| | SUB |
| | MULT |
| | DIV |
| 2) Operações de transferência: | CAR |
| | LEVE |
| | FAÇA |
| | CSEC |
| | TROC |
| | NEG |
| 3) Operações de desvio: | VA |
| | TESTZ |
| | COMP |
| 4) Operações de movimento: | MOVE |
| | MOVD |
| | RODE |
| | RODD |
| 5) Operações de entrada/saída: | NPED |
| | CPED |
| | NIMP |
| | CIMP |

b) Campo de Operando.

O significado e formato de um campo de operando depende do tipo de código de operação usado no comando fonte.

Um campo de operando pode ser:

- a) um nome de registrador - com uma das letras A, B, C, D ou P;

- b) um termo simbólico ou numérico, ou combinações de termos simbólicos, numéricos e operadores +, -, *, /, ou antecidos de '.' (endereço relativo ao contador de endereços) ou @ (endereçamento indireto).

Termo Simbólico - pode ser constituído de letras de A a Z e dígitos de 0 a 9, começando com letra, de comprimento máximo de 8 caracteres. Um símbolo usado num campo de operando deve ter sido definido anteriormente no programa.

Termo Numérico - números inteiros positivos.

Ponto - um ponto no campo de operando se refere ao valor no registrador P, ao tempo em que o comando fonte é encontrado.

Arroba - o símbolo @ indica que o endereçamento é indireto.

Expressões - símbolos e números podem ser ligados por operadores aritméticos para formar expressões aritméticas de endereçamento.

Exemplos de Instruções:

```
LEVE  A  100
FAÇA  D  . + 5
SOME  C  INC1 - 3
CAR   P  @241
```

3.4.3. Pseudo-Instrução.

A pseudo-operação " NOME 'símbolo' . 'endereço' ", dá a um símbolo um valor geralmente diferente do valor atual do registrador P. Esta operação nomeia posições de memória.

Ao símbolo do 1º operando é dado o valor representado no

2º operando. O campo do 2º operando pode conter qualquer expressão. O valor do 2º operando deve ser um número entre 1 e 500 ou um símbolo previamente definido no programa fonte.

Este comando não resulta numa instrução de máquina.

Exemplos:

NOME CONST 50 - a partir desta instrução, qualquer referência ao termo simbólico CONST é uma referência à posição 50 de memória.

NOME INC CONST-3 - a partir desta instrução, qualquer referência a INC, significa referência à locação 47, se CONST foi definido como acima.

3.4.4. Formato de um programa em Linguagem 'Assembler'.

Toda instrução 'assembler' corresponde a uma instrução em linguagem de máquina e é traduzida por essa instrução.

As pseudo-instruções não correspondem a instruções em linguagem de máquina; são operações que fornecem informação ao montador 'assembler'. Exemplo: NOME.

3.4.5. Especificações das Instruções.

Na multiplicação, a soma do comprimento do multiplicando pelo comprimento do multiplicador não deve exceder de 8, a menos de truncamento à direita.

Na divisão, a parte decimal é truncada, pois só são permitidos números inteiros.

3.4.6. Comentários.

Comentários podem aparecer ao fim de um comando, a partir da coluna 51, antecidos pelo símbolo '/ '.

Exemplo: CIMP A / Imprime caracter

Um comentário não é traduzido para linguagem de máquina. Sua utilidade é dar maior clareza a um programa.

3.5. Comandos da Linguagem do Interpretador.

\$INICIE ou \$INIC - inicia sessão.

\$TERMINE ou \$TERM - termina sessão.

\$EXECUTE ou \$EXEC - inicia execução.

\$PARE EXECUÇÃO ou \$PEXEC - interrompe a execução do programa.

\$CONTINUE ou \$CONT - continua a execução do programa após parada, do ponto do programa onde a parada foi exigida.

\$APAGA CHARACTER ou \$APCAR - apaga último caracter da linha.

\$APAGA PALAVRA ou \$APPAL - apaga última palavra da linha.

\$APAGA LINHA ou \$APLIN - apaga última linha.

\$REFORMA ou \$REF - reimprime a linha, ordenando, i.e., eliminando os brancos dos caracteres apagados.

~~\$ENTRADA~~ ou ~~\$ENTR~~ - ~~avisa término de linha e imprime o comando se é~~ comando em linguagem 'assembler'.

\$RETORNO ou \$RETR - avisa término de linha.

\$APAGA MEMÓRIA ou \$APMEM - apaga todos os conteúdos da memória (obs.: os que possuem valor).

\$APAGA MEMÓRIA n ou \$APMEM n - apaga o conteúdo da posição n de memória.

\$APAGA REGISTRADOR ou \$APREG - apaga o conteúdo de todos os registradores.

\$APAGA REGISTRADOR r ou \$APREG r - apaga o conteúdo do registrador r.

\$LISTA MEMÓRIA ou \$LISTM - lista todas as posições de memória (endereço e conteúdo) que possuem valor.

\$LISTA MEMÓRIA n ou \$LISTM n - lista o endereço e o conteúdo da posição de memória n.

\$LISTA REGISTRADOR ou \$LISTR - lista endereço e conteúdo dos registradores que possuem valor.

\$LISTA REGISTRADOR r ou \$LISTR r - lista endereço e conteúdo do registrador r.

\$FIXA MEMORIA n ou \$FIX n - fixa a posição de memória n, permitindo alteração do seu conteúdo.

\$TRANSFERE n,m ou \$TRANS n,m - transfere o conteúdo da posição n de memória para a posição m.

\$TRANSFERE n/,m ou \$TRANS n/,m - transfere o conteúdo de todas as posições de memória a partir de n (inclusive) para as posições a partir de m (inclusive).

\$TRANSFERE n/p,m ou \$TRANS n/p,m - transfere o conteúdo das posições de memória de n até p (inclusive) para as posições a partir de m (inclusive).

\$NOMES - lista todos os nomes simbólicos definidos e endereços correspondentes.

Obs.: Todos os comandos das várias linguagens, devem ser seguidos do comando \$RETORNO ou \$ENTRADA.

Características de Depuração do Mic-Mac.

- a) um comando de 'assembler', seguido de \$ENTRADA, fornecerá o comando em linguagem assembler.
- b) o comando \$FIX n, seguido do comando \$ENTRADA, fornece o conteúdo da posição n, para que se tenha certeza da instrução a ser modificada.

- c) o comando \$LISTM seguido do comando \$ENTRADA, listará as instruções em linguagem 'assembler' e de máquina.
- d) o comando \$EXEC seguido do comando \$ENTRADA, fará com que o conteúdo dos registradores e a instrução atual sejam impressos a medida que o programa é executado.

- operações;
- ciclos;
- ramificação;
- procedimentos;
- recursão e iteração;
- formalização de problemas.

4.3. Estruturação do Currículo.

4.3.1. Determinação das Unidades de Ensino.

- 1 - Relacionamento Homem-Máquina.
- 2 - Introdução à utilização da máquina.
- 3 - Descrição geral da máquina.
- 4 - Distinção nome-valor.
- 5 - Noção de programa.
- 6 - Alteração de sequência de execução.
- 7 - Instruções aritméticas.
- 8 - Distinção entre comandos do interpretador e do 'assembler'.
- 9 - Operações de Entrada e Saída.
- 10 - Armazenamento de dados.
- 11 - Nomes simbólicos.
- 12 - Tomada de decisões.
- 13 - Operações de movimento.
- 14 - Endereçamento relativo.
- 15 - Endereçamento indireto.
- 16 - Procedimentos.
- 17 - Regras de parada em procedimentos.
- 18 - Conjunto de instruções.
- 19 - Aplicação final.

4.3.2. Determinação da sequência das unidades.

Para este curso, a determinação da sequência das unidades e dos tópicos em cada unidade será pré-estabelecida.

O currículo será estruturado e a apresentação das lições obedecerá à sequência estabelecida no currículo, à exceção de retrocessos exigidos pela atuação do aluno.

Desta forma, o estudante não terá acesso ao currículo, i.e., à determinação própria da sequência de apresentação de conceitos.

4.3.3. Descrição das Unidades de Ensino.

Lição 1 - RELACIONAMENTO HOMEM-MÁQUINA.

Objetivos: Fazer com que os estudantes

- conheçam a utilidade do computador;
- entendam como o homem se comunica com a máquina.

Conteúdo: - Breve histórico da evolução do computador.
- Mostrar que a comunicação é feita através de linguagens, utilizando-se da máquina de escrever, TV ou papel.

Lição 2 - INTRODUÇÃO À UTILIZAÇÃO DA MÁQUINA.

Objetivos: Fazer com que os estudantes identifiquem e saibam utilizar os comandos de edição relativos ao terminal.

Conteúdo: Introduzir:

- comandos \$INICIE e \$TERMINE;
- comandos \$APAGA CHARACTER, \$APAGA

PALAVRA e \$APAGA LINHA;

- comandos \$RETORNO, \$ENTRADA e \$REFORMA.

Lição 3 - DESCRIÇÃO GERAL DA MÁQUINA.

Objetivos: Fazer com que os estudantes identifiquem os conceitos de:

- memória;
- memória limitada;
- registradores;
- contador de endereços.

Conteúdo: Introduzir conceitos de:

- memória e tamanho de memória;
- registradores: definição e utilização;
- registrador P.

Lição 4 - DISTINÇÃO NOME-VALOR.

Objetivos: Fazer com que os estudantes

- façam a distinção entre nome-valor;
- reconheçam literais ou valores como números decimais em instruções.

Conteúdo: Mostrar como valores ou números decimais são armazenados - representação de sinal;

Introduzir comandos \$LISTM e \$LISTR e sua associação ao comando \$ENTRADA;
Introduzir comando FAÇA, enfatizando a relação nome e valor das instruções.

Lição 5 - NOÇÃO DE PROGRAMA.

Objetivos: Fazer com que os alunos identifiquem

- programa;
- execução de programas;
- armazenamento de programas;
- comandos de controle para ordenar execução, parada e continuação.

Conteúdo: - Dar a noção de programa - exemplos e comparações reais;

- Introduzir armazenamento de programas;
- Introduzir execução sequencial de programas;
- Introduzir comandos \$EXEC, \$PEXEC, \$CONT;
- Introduzir comando PARE.

Lição 6 - ALTERAÇÃO DE SEQUÊNCIA DE EXECUÇÃO.

Objetivos: Fazer com que os estudantes:

- façam distinção entre sequência natural e alterada da execução de um programa;
- identifiquem o comando para alterar a memória.

Conteúdo: - Introduzir alteração de sequência de execução pelo comando FAÇA P n;

- Relacionar FAÇA e VA;
- Introduzir noção de ciclo;
- Introduzir comando \$FIX e enfatizar a utilidade de tal comando associado

ao comando \$ENTRADA;

- Introduzir comandos \$APMEM e \$APREG.

Lição 7 - INSTRUÇÕES ARITMÉTICAS.

Objetivos: Fazer com que os estudantes:

- entendam a capacidade de um computador executar operações;
- identifiquem os três campos de uma instrução;
- distingam instruções com campo de valor, de instruções com campo de endereço.

- Conteúdo:
- Introduzir comandos SOME, SUB, MULT e DIV;
 - Introduzir representação de instruções em três campos - 'assembler' e máquina;
 - Enfatizar a diferença de instruções quanto ao formato;
 - Introduzir conceitos de 'overflow' e truncamento.

Lição 8 - DISTINÇÃO ENTRE COMANDOS DO INTERPRETADOR E DO 'ASSEMBLER'.

Objetivos: Fazer com que os estudantes:

- entendam o contexto computacional, fazendo a distinção dos três níveis da estrutura do MIC-MAC;
- façam a distinção entre comandos do interpretador, comandos do 'assembler'

e dados.

- Conteúdo:
- Explicar a estrutura funcional do MIC-MAC;
 - Contrastar a sintaxe e semântica dos comandos do interpretador e as instruções do 'assembler';
 - Introduzir 'comentários'.

Lição 9 - OPERAÇÕES DE ENTRADA E SAÍDA.

Objetivos: Fazer com que os estudantes:

- saibam fornecer dados durante a execução;
- entendam os conceitos de função, domínio, funções totais e parciais, com posição e inversa de funções.

- Conteúdo:
- Introduzir comando NPED - enfatizar 'overflow' e truncamento;
 - Introduzir conceito de função através do comando CPED;
 - Introduzir o conceito de inversa pelo comando CIMP;
 - Introduzir comando NIMP;
 - Introduzir conceito de composição, do domínio de funções, funções parciais e totais.

Lição 10 - ARMAZENAMENTO DE DADOS.

Objetivos: Fazer com que os estudantes:

- entendam a necessidade de se armaze-

nar certos dados;

- reconheçam uma melhor utilização de registradores;
- entendam a necessidade da separação de armazenamento de programa-dados.

Conteúdo: - Introduzir comandos CAR e LEVE;

- Enfatizar a direção oposta na execução dos dois comandos;
- Enfatizar operações realizadas em registradores e o perigo da destruição de resultados.

Lição 11 - NOMES SIMBÓLICOS.

Objetivos: Fazer com que os estudantes:

- saibam dar nomes simbólicos apropriados a locações de memória;
- façam distinção entre instrução e pseudo-instrução do 'assembler'.

Conteúdo: - Introduzir pseudo-instrução NOME;

- Definir nomes simbólicos;
- Introduzir o comando de edição \$NOMES.

Lição 12 - TOMADA DE DECISÕES.

Objetivos: Fazer com que os estudantes:

- identifiquem 'tomada de decisão';
- entendam a necessidade de um programa tomar decisões;
- identifiquem a noção de erro e depuração.

- Conteúdo:
- Introduzir o comando TESTZ, explicando a necessidade de se tomar decisões;
 - Dar exemplos de programas com erros ('grilos') e explicar o meio de descobrir tais erros pela noção de depuração;
 - Enfatizar os comandos \$PEXEC e \$CONT;
 - Introduzir o comando CSEC.

Lição 13 - OPERAÇÕES DE MOVIMENTO.

Objetivos: Fazer com que os alunos:

- entendam a noção de 'palavra' para o computador;
- identifiquem o modo de transmitir palavras ao computador através dos comandos de movimento.

- Conteúdo:
- Introduzir os comandos MOVE, MOVD, RODE, RODD;
 - Explicar como se processam palavras no computador e dar exemplos de programas para tal.

Lição 14 - ENDEREÇAMENTO RELATIVO.

Objetivos: Fazer com que os estudantes:

- identifiquem endereçamento relativo;
- entendam o significado e a utilidade do endereçamento relativo.

- Conteúdo: - Introduzir endereçamento relativo (em relação ao registrador P e a nomes simbólicos) e aplicá-lo para reformulação de programas anteriores;
- Introduzir o comando de edição \$TRANSFER.

Lição 15 - ENDEREÇAMENTO INDIRETO.

Objetivos: Fazer com que os alunos:

- identifiquem endereçamento indireto num programa;
- entendam a utilidade do endereçamento indireto;
- entendam o funcionamento de endereçamento indireto.

- Conteúdo: - Introduzir o conceito de endereçamento indireto;
- Explicar a utilidade de endereçamento indireto;
 - Mostrar que o significado do dado depende de como e porque está sendo usado;
 - Esclarecer a diferença entre endereços, programa-dados separados;
 - Introduzir a estrutura de vetor;
 - Mostrar a representação de endereçamento indireto em linguagem de máquina.

Lição 16 - PROCEDIMENTOS.

Objetivos: Fazer com que os alunos:

- identifiquem procedimentos;
- saibam chamar 'procedimentos', retornar deles e passar argumentos.

Conteúdo: - Introduzir o conceito de 'procedimento' e enfatizar sua utilidade;

- Dar regras de chamada e passagem de argumentos;
- Introduzir a operação NADA.

Lição 17 - REGRAS DE PARADA EM PROCEDIMENTOS.

Objetivos: Fazer com que os alunos:

- identifiquem o melhor meio de parada de acordo com o problema;
- entendam 'controle de programa'.

Conteúdo: - Introduzir comandos COMP e NEG;

- Dar diferentes regras de parada em procedimentos iterativos;
- Introduzir o comando TROC e instruções com dois campos de registrador.

Lição 18 - CONJUNTO DE INSTRUÇÕES DO ASSEMBLER E DO INTERPRETADOR.

Objetivos: Fazer com que os estudantes:

- revejam todas as instruções aprendidas;
- distingam as diversas linguagens;
- distingam o formato das diferentes instruções do 'assembler'.

- Conteúdo:
- Dar o conjunto de instruções do 'assembler';
 - Dar o conjunto de instruções do interpretador;
 - Dar instruções grupadas por formato;
 - Dar pequenos programas englobando todas as instruções.

Lição 19 - APLICAÇÃO FINAL.

Objetivos: Fazer com que os estudantes:

- utilizem todas as instruções;
- identifiquem o inter-relacionamento de 'procedimentos'.

Conteúdo:

- Dar programas englobando todas as instruções e reunindo programas anteriores.

4.4. Estruturação de uma Lição.

4.4.1. Exemplo de Lição subdividida em Tópicos.

Lição 9: Operações de entrada e saída.

Tópico 1 - Definição de dados.

Tópico 2 - Comando NPED - representação de sinal, overflow e truncamento.

Tópico 3 - Comando CPED - códigos.

Tópico 4 - Noção de função e domínio - funções parciais (NPED) e totais.

Tópico 5 - Noção de inversa e composição de funções e simetria.

Tópico 6 - Prática dos comandos de entrada e saída,
através de programas-exemplos e progra-
mas-exercícios.

4.4.2. Exemplo da Elaboração de um Tópico.

Exemplo do conteúdo do Tópico 3 da Lição 9, em função do mo-
delo (capítulo 2).

COMANDO CPED

Conceito e/ou exemplo	===== Você agora vai aprender um novo literal: o <u>ca</u> racter. Um caracter significa um símbolo que será en- tendido pela máquina. Como é que o computador vai saber o que é um símbolo, se ele só pode guardar números em sua memória? Por esta razão, cada símbolo possui um código. Pegue o seu manual e abra na página do código MM. =====
Questão 1	O código do computador para a letra 'A' é 60. Qual o código para a letra 'P'? (75)
Questão 2	O código 45 corresponde ao símbolo '?'. . Qual o símbolo que corresponde ao código 46? (Q)
Questão 3	Os algarismos também podem ser representados como caracteres.

	Qual o código para o algarismo 7? (57)
	Você pode observar que o número de código para o algarismo 7 é diferente do próprio algarismo 7. Isto serve para fazer a diferença entre um <u>n</u>úmero representado como dado numérico e um <u>n</u>úmero representado como caracter.
Questão 4	Um código MM é um número composto de ____ algarismos. (2)
Questão 5	Qual o comando que você aprendeu há pouco, que serve para fornecer dados numéricos a um programa? (NPED)
Questão 6	Existe também um comando para fornecer caracteres ao computador: o <u>comando CPED</u>. CPED é a abreviação para 'Caracter pedido'; você poderá usar qualquer um dos dois. Qual o comando usado para fornecer caracteres ao computador? (CPED)
Questão 7	O comando NPED A faz com que o dado numérico escrito fosse colocado internamente no registrador _____. (A) O comando CPED também é seguido do nome ou <u>n</u>úmero de um registrador, da mesma forma que o comando NPED.

	Exemplo: CPED B
Questão 8	Responda Sim ou Não: Você pode colocar dados diretamente na memória do computador MIC-MAC? (Não)
Questão 9	Tanto para o comando NPED como para o comando CPED, os dados pedidos são colocados nos _____. (registradores)
Questão 10	Você fornecia o dado numérico pedido pelo comando _____ quando o programa estava em _____. (NPED, execução) O mesmo acontece para o comando CPED. Quando o comando CPED estiver sendo executado, será impresso: 'CHARACTER ?'. Neste momento você vai fornecer o caracter que desejar.
Questão 11	Vamos tentar? Escreva \$APREG para apagar tudo o que estiver contido nos registradores do computador. Não se esqueça do comando \$RETORNO ou \$RETR a pós cada comando e após o dado. Escreva CPED A e o comando \$EXEC e, quando aparecer a impressão 'CHARACTER ?', forneça a letra 'X', e depois o comando \$LISTR A. Muito bem. Você forneceu ao computador um caracter, colocando-o no registrador A. O que foi impresso depois do comando \$LISTR A? Exatamente o código da letra X. Verifique no manual.

	Agora escreva o comando \$LISTR seguido do comando \$ENTRADA. Você verá a representação <u>in</u> terna da letra X e a própria letra X.
Questão 12	Agora você sabe que existe uma diferença entre o comando NPED e o comando CPED. O comando NPED faz com que o dado numérico fornecido do meio externo seja colocado no <u>re</u> gistrador indicado na instrução.
	O comando CPED faz com que o _____.do <u>carac</u> ter fornecido seja colocado no registrador <u>in</u> dicado na instrução. (código)
Questão 13	Vamos então ver a diferença com números. Se escrevemos o algarismo 5 em resposta ao <u>co</u> mando NPED B, o registrador B conterá o <u>núme</u> ro _____. (5) Se escrevemos o algarismo 5 em resposta ao <u>co</u> mando CPED B, o registrador B irá conter o número _____. (55)
	Você agora pode testar, escrevendo um <u>progra</u> minha com NPED A e CPED B e fornecer o <u>mes</u> mo número para os dois registradores.
Questão 14	Quantos dígitos possui um registrador? (8)
	Cada caracter será colocado no registrador na forma de seu código, não é?
Questão 15	Todo código tem dois algarismos.
	Então, estes dois algarismos serão colocados

	à direita do registrador, e os demais dígitos serão preenchidos com zeros. Exemplo: 'o caracter 'Q' será representado internamente num registrador como 00000076.
	Qual a representação interna do caracter 'B'? (00000061)
Questão 16	Qual a representação interna do caracter ', ' (vírgula)? (00000003)
Questão 17	A que caracter corresponde a representação interna 00000089 do registrador A? ('\$')
Questão 18	Assim vemos que, pelo comando CPED, cada registrador só pode conter ____ caracter. (1) E como podemos escrever palavras para o computador? Isto você aprenderá mais tarde, utilizando o comando CPED e outros.

PERGUNTAS SIMILARES

Questão 1	1 - Qual o código para o símbolo ';'? (04) 2 - Qual o código para a letra 'O'? (74) 3 - Qual o código para o símbolo '+'? (96) 4 - Qual o código para a letra 'V'? (81)
-----------	---

Questão 2	1 - Qual o símbolo que corresponde ao código 85? ('Z') 2 - Qual o símbolo que corresponde ao código 84? ('Y') 3 - Qual o símbolo que corresponde ao código 93? (' (') 4 - Qual o símbolo que corresponde ao código 02? (' . ')
Questão 3	1 - Qual o código para o algarismo 8? . (58) 2 - Qual o símbolo que corresponde ao código 65? ('F') 3 - Qual o código para o algarismo 7? (57) 4 - Qual o caracter que corresponde ao código 61? ('B')
Questão 4	1 - O símbolo 'P' possui código _____, composto de _____ algarismos. (75,2) 2 - O símbolo _____ possui código 69. Todo código é composto de _____ algarismos. (J, 2)
Questão 5	1 - O comando NPED é um comando da linguagem _____ do MIC-MAC, que é usado num programa para pedir dados _____ do meio externo. (assembler, numéricos) 2 - O comando NPED A fará com que um dado _____ seja pedido do meio _____ e armazenado no _____. (numérico, externo, registrador A) 3 - Qual o comando que servirá para fornecer um dado numérico para ser armazenado no registrador P? (NPED P)

Questão 6	1 - O comando CPED serve para fornecer _____ ao computador. (caracteres) 2 - Responda Sim ou Não: O comando CPED da linguagem 'assembler' serve para fornecer dados numéricos ao computador? (Não) 3 - Que tipo de dado é requerido por um comando CPED? (caracter)
Questão 7	1 - Onde será armazenado o dado numérico a ser fornecido, requerido pelo comando NPED 'B'? (registrador B) 2 - Qual o comando que fará com que o dado numérico seja fornecido do meio externo e armazenado no registrador C? (NPED C) 3 - Qual o comando que fará com que um caracter seja fornecido do meio externo e armazenado no registrador D? (CPED D)
Questão 8	1 - O comando NPED 100 está correto? (Não) Ele não está correto porque dados fornecidos do meio externo não podem ser armazenados diretamente na _____ do MIC-MAC. (memória) 2 - Um dado fornecido do meio externo só pode ser armazenado num _____. (registrador)
Questão 9	1 - Um registrador é usado para armazenar _____ vindos do meio externo ou executar _____. (dados, cálculos ou operações) 2 - Os comandos NPED e CPED indicam Número _____ e _____, a serem armazenados em _____. (pedido, caracter, pedido, registradores)

Questão 10

- 1 - Responda Sim / Não a cada uma das perguntas:
- a) Um dado numérico ou caracter, pedido pelo comando NPED ou CPED, é fornecido no momento em que o programa está sendo escrito? (Não)
 - b) Um dado numérico é fornecido antes de ser escrito o comando \$EXEC? (Não)
 - c) Um caracter deverá ser fornecido no momento em que um comando CPED estiver sendo executado? (Sim)
- 2 - Um dado numérico é exigido pelo comando _____ e um caracter é exigido pelo comando _____, no momento em que esses comandos estiverem sendo _____.
- (NPED, CPED, executados)

Questão 11

- 1 - Escreva Verdadeiro ou Falso às assertivas abaixo:
- a) O comando \$APMEM faz com que o conteúdo de todos os registradores sejam apagados. (F)
 - b) O comando \$APREG A faz com que o conteúdo do registrador A seja apagado. (V)
 - c) O comando \$EXEC faz com que um programa seja executado. (V)
 - d) O comando CPED pede que um dado numérico seja fornecido ao computador. (F)
- Faça com que o caracter '=' seja colocado no registrador D.
- 2 - Faça com que o caracter ';' seja colocado no registrador C.

Questão 12	1 - Responda Sim ou Não a cada uma das perguntas abaixo: a) Um dado numérico será colocado exatamente como é, a menos de sinal, no registrador indicado no comando NPED? (Sim) b) O código do carácter fornecido será colocado no registrador indicado no comando CPED? (Sim) 2 - O que é colocado no registrador C, se o símbolo 'L' é fornecido, ao ser pedido pelo comando CPED C? (código de 'L' ou 71) 3 - O que é colocado no registrador B se o símbolo ':' é fornecido em resposta ao comando CPED B? (05)
Questão 13	1 - Se o número 8 foi colocado no registrador C, o comando correspondente foi _____. (NPED C) Se o código 55 foi colocado no registrador A, o comando correspondente foi _____ e o carácter fornecido foi _____. (CPED A, 5) 2 - O que será colocado no registrador C, se se fornece o algarismo 9 em resposta ao comando CPED C? (59)
Questão 14	1 - Um registrador é utilizado para fazer _____. Ele também é utilizado para armazenar dados quando fornecidos durante a _____ de um programa. O conteúdo de um registrador será composto de _____ algarismos. (cálculos ou operações, execução, 8) 2 - O maior número que pode ser colocado em um registrador deve ter no máximo _____ algarismos, pois um dígito é guardado para o _____

	_____. Logo um registrador tem _____ dígitos. (7, sinal, 8)
Questão 15	1 - Qual a representação interna do caracter 'C'? (00000062) 2 - Qual a representação interna do caracter 'O'? (00000050) 3 - Qual a representação interna do caracter 'E'? (00000064)
Questão 16	1 - Qual a representação interna do símbolo ':'? (00000005) 2 - Qual a representação interna do caracter ';'? (00000004) 3 - Qual a representação interna do símbolo '-'? (00000097)
Questão 17	1 - A que caracter corresponde a representação interna 00000082? ('W') 2 - A que caracter corresponde a representação interna 00000094? ('I') 3 - A que caracter corresponde a representação interna 00000066? ('G')
Questão 18	1 - Cada caracter fornecido será colocado nos _____ dígitos mais a _____ do registrador indicado no comando CPED. (2, direita) 2 - Responda Sim / Não a cada uma das perguntas: a) Dois caracteres podem ser colocados ao mesmo tempo num registrador pelo mesmo comando CPED? (Não) b) Um número de três algarismos pode ser colocado num registrador por um comando CPED? (Não)

- c) Cada caracter será armazenado através de seu código? (Sim)
 - d) Um código será armazenado na memória pelo comando CPED? (Não)
- =====

4.5. Exemplo de Exercícios.

Uma vez que o tópico 3 da lição 9 foi exemplificado, daremos um exemplo de exercícios para a avaliação de tal tópico. Estes exercícios seriam correspondentes ao primeiro semigrupo do nível básico mínimo.

Exemplo:

Para cada um dos itens abaixo, indique a letra correspondente à resposta da questão:

1 - A letra 'F' para o computador é _____.

- a) um número.
- b) um caracter ou um número.
- c) um registrador.
- d) um caracter.
- e) nenhuma das acima.

2 - Um código MM é um _____ que será a representação interna de um símbolo.

- a) caracter.
- b) número de dois algarismos.
- c) número de um algarismo.
- d) algarismo.
- e) nenhuma das acima.

3 - Se o símbolo '=' for fornecido em resposta a um comando NPED, o computador _____ .

- a) guardará o código do símbolo no registrador indicado.
- b) guardará o símbolo no registrador indicado.
- c) dará uma mensagem de erro.
- d) colocará o código de tal símbolo numa posição da memória.
- e) nenhuma das acima.

4 - O código correspondente ao algarismo 5 é _____ .

- a) o número 32.
- b) o número 5.
- c) o número 48.
- d) o número 93.
- e) nenhuma das acima.

5 - O comando para fornecer o número 9, durante a execução de um programa, que será armazenado no registrador C é _____ .

- a) FAÇA C 9.
- b) CPED C:
- c) NPED 9.
- d) NPED C.
- e) nenhuma das acima.

6 - O comando CPED B está sendo executado.

Você escreve o símbolo 'I', seguido do comando \$LISTR B.

O conteúdo do registrador B será _____ .

- a) 68000000.
- b) 00000022.
- c) 00000068.
- d) I.
- e) nenhuma das acima.

a) Gerais.

Todas as instruções da linguagem do interpretador e do 'assembler' serão ensinadas no decorrer do curso.

A linguagem de máquina não será ensinada em todos os detalhes. Ela deverá ser usada sempre que necessária para facilitar a compreensão de certos conceitos, como, por exemplo:

- máquinas só entendem números;
- formato de instruções;
- possibilidade de erros na má distribuição de programa-dados, etc.

As características de depuração do MIC-MAC devem ser exploradas, visto que permitem maior clareza na distinção das linguagens 'assembler' e máquina e possibilitam visualização da execução dos programas.

Fazer com que os alunos tenham uma boa visão da capacidade do computador (que o distingue das demais máquinas) que é:

- decidir;
- armazenar informação;
- guardar programas na memória.

b) Específicas.

Lição 1 - Dar um breve histórico da evolução do computador, com exemplos práticos de sua utilização (indústria e escola) incluindo as pesquisas recentes (jogos, etc.)

Lição 2 - Motivar os estudantes a se comunicarem com a máquina, encorajando-os a escrever qualquer coisa e observarem as respostas do computador. Estes pequenos diálogos servirão para uma análise da criatividade de cada estudante relativa à comunicação com a máquina.

Dar ênfase a que os alunos entendam porque as máquinas não entendem a linguagem humana (uma vez que os pró-

prios humanos não a entendem) - relacionar a pessoas que falam línguas diferentes e a dificuldade de comunicação entre pessoas de mesma língua, dada a extensão de vocabulário.

Introduzir os comandos de edição com aplicação imediata.

Lição 3 - Relacionar a memória do computador à nossa memória.

Estimular a compreensão de limite, provocando erros.

Exemplo: FAÇA P 1000.

Cada conceito deverá ser relacionado a fatos reais para melhor compreensão.

Lição 4 - Enfatizar que as máquinas só entendem números.

Mostrar o relacionamento da instrução 'assembler' e código de máquina, registradores e números correspondentes em máquina.

Enfatizar endereçamento e conteúdo de memória.

Enfatizar valor na instrução FAÇA.

Introduzir comando 'assembler' seguido de \$ENTRADA.

Lição 5 - Relacionar programas de computação a programas da vida cotidiana, como 'sequência de instruções que alguém ou alguma máquina possa entender e obedecer'.

Noção de programa como lista de valores executados por uma máquina, para a qual estes valores têm significado.

Lição 6 - Motivar a execução sequencial das instruções.

Introduzir ciclo através do comando: '05: FAÇA P 5'

Mostrar a possibilidade da descoberta de erros através das características de depuração do MIC-MAC e correção através do comando \$FIX.

Lição 7 - Através da linguagem de máquina, mostrar a divisão dos três campos de uma instrução.

Relacionar instruções com valor e instruções com endereço.

Enfatizar como os registradores são utilizados para fazer cálculos.

Introduzir conceitos de 'overflow' e truncamento, utilizando as operações aritméticas com dados que ocasionem tais problemas.

Lição 8 - Procurar simbolizar os níveis da estrutura do MIC-MAC com relacionamentos que permitam fácil visualização e compreensão.

Explicar a distinção da execução de instruções do interpretador (execução imediata) e 'assembler' (traduzido e guardado para execução após o comando \$EXEC).

Lição 9 - Explicitar códigos e citar manual.

Enfatizar o novo literal - o caracter teclado.

Introduzir conceito de função, utilizando entrada (caracter) e saída (código de caracter) - domínio = conjunto de símbolos.

Conceito de simetria - mostrar porque CPED e CIMP não podem ser usados ao contrário.

Conceito de inversa e composição de funções pelos comandos CPED e CIMP.

Conceito de funções parciais pelo comando NPED em relação ao domínio dos caracteres.

Lição 10 - ~~Mostrar como~~ um valor pode ser destruído, utilizando NPED ou CPED duas vezes seguidas para o mesmo registrador, mostrando a necessidade do armazenamento na memória.

Enfatizar a possibilidade de destruição de um programa, armazenando dados onde existem instruções a serem utilizadas.

Lição 11 - Mostrar como um nome pode ser escolhido para refletir o conteúdo numa posição de memória, tornando-se fácil

lembrar o seu nome-valor.

Lição 12 - Relacionar 'tomadas de decisão' a exemplos simples da vida real.

Dar um exemplo de programa com erro, para introduzir a noção de 'grilo' num programa. Isto oferece um bom teste para se verificar a habilidade do estudante em predizer o que um programa dado fará.

Explicar depuração, relacionando-a ao papel do detetive, i.e., o estudante finge ser o computador.

Lição 13 - Enfatizar como cada caracter ocupará uma palavra, se fornecido pelo comando CPED.

Exemplificar palavras de menos de cinco letras, sendo fornecidas como dados do próprio programa e ensinar os comandos de movimento, utilizando-os para imprimir palavras.

Lição 14 - Mostrar que o endereçamento relativo é útil para os comandos CAR e LEVE, para valores de ou para posições de memória, cujos endereços são calculados pelo programa em execução.

Mostrar o uso de expressões aritméticas no campo de endereço duma instrução.

Lição 15 - Neste momento pode ser mostrado o dígito de endereçamento indireto em máquina (ignorado até então) e o símbolo @ na linguagem 'assembler'.

Dar a idéia principal do endereçamento indireto, supondo que a memória tenha mais de 10.000 palavras, relacionando ao tamanho limite de endereço em uma instrução.

Lição 16 - Mostrar como o endereçamento indireto pode ser usado para procedimentos.

Exemplificar e dizer o porquê de 'procedimentos', e a possibilidade de execução de procedimentos com dados diferentes.

Lição 17 - Relembrar TESTZ e relacioná-lo a COMP como instruções usadas para tomada de decisão.

Enfatizar formato de instruções de dois registradores e compará-las aos outros formatos.

Lição 18 - Listar todas as instruções e exemplificar com programas dos mais simples aos mais complicados.

Lição 19 - Relembrar as características básicas do computador: conjunto de instruções, memória acessível, execução controlada.

Enfatizar a diferença entre computadores e máquinas comuns: capacidade de armazenar programas, lembrar coisas e tomar decisões.

4.7. Exemplo de Comentário-Resposta e Ajuda Automática.

Assunto: Endereçamento indireto.

Exercício:

De acordo com os comandos abaixo, complete a sentença.

NOME	LOCAL	10
FAÇA	A	20
LEVE	A	10
FAÇA	B	5
LEVE	B	@LOCAL

O último comando fará com que o valor 5 seja colocado no endereço _____

_____ . (20)

(acertou) . Excelente.

O símbolo @ indica endereçamento indireto.

Isto quer dizer que o endereço onde será armazenado o valor 5 não é LOCAL, mas o conteúdo do endereço denominado LOCAL.

(errou) . Você tem certeza?

Vamos relembrar alguns conceitos.

Daremos uns exemplos para que você verifique porque se enganou.

- . FAÇA D 8 coloca o valor 8 como conteúdo do registrador D.
- . LEVE A 5 coloca o conteúdo do registrador A no endereço 5.
- . LEVE C @5 coloca o conteúdo de C no endereço dado pelo conteúdo da posição de memória 5.
- . FAÇA B 101
- LEVE B 5
- LEVE C @5 coloca o conteúdo de C no endereço 101 que é o conteúdo da posição 5 de memória.

OK? Agora tente novamente.

4.8. Sugestão para o material do curso.

Sugerimos que seja preparado:

- 1) um manual de lições - este manual deve conter todas as lições, se possível com ilustrações. Os alunos podem receber este manual no início do curso ou receber as lições separadamente, na medida em que as mesmas sejam efetuadas.
- 2) um manual de exercícios - este manual deve conter todos os exercícios. Os exercícios efetuados na sessão devem ser entregues aos alunos ao término da mesma.
- 3) um manual para consulta imediata - todos os alunos devem receber tal manual no início do curso. Ele deve conter:
 - breve descrição da sintaxe e semântica da linguagem 'assembler';
 - breve descrição das características da máquina;
 - lista de instruções do 'assembler' com os respectivos códigos e breve descrição;
 - lista de comandos do interpretador e suas funções;
 - registradores de trabalho;
 - código MM.

APÊNDICE A-1CONJUNTO DE INSTRUÇÕES DO MIC-MAC

- CAR ou CR - Carrega o registrador com o conteúdo da memória indicada.
- CIMP ou CIP - (Caracter a ser impresso) - Imprime um dado alfanumérico, correspondente ao código MM dos dois dígitos de mais baixa ordem do registrador indicado.
- COMP ou COP - Compara os conteúdos dos registradores indicados; se diferentes, a próxima instrução é pulada.
- CPED ou CPD - (Caracter pedido) - Requer um dado alfanumérico, cujo código MM correspondente será colocado nos dois dígitos de mais baixa ordem do registrador indicado.
- CSEC ou CSC - (Consecutivo) - Incrementa de 1 o conteúdo do registrador indicado.
- DIV ou DV - Divide o conteúdo do registrador indicado pelo conteúdo da memória indicada.
- FAÇA ou FÇ - Faz o conteúdo do registrador indicado igual ao valor.
- LEVE ou LEV - Leva o conteúdo do registrador indicado para a posição de memória indicada.
- MOVE ou MVE - (Move esquerda) - Move os dígitos do registrador indicado para a esquerda, tantas vezes quantas indique o valor. Os dígitos de mais alta ordem são perdidos.
- MOVD ou MVD - (Move direita) - Move os dígitos do registrador indicado para a direita tantas vezes quantas indique o valor. Os dígitos de mais baixa ordem são perdidos.
- MULT ou MT - Multiplica ao registrador indicado o conteúdo da memória indicada. O resultado permanece no registrador.
- NADA ou ND - Não faz operação alguma.
- NEG - Nega o conteúdo do registrador indicado.
- NIMP ou NIP - (Número a ser impresso) - Imprime um dado numérico, conteúdo do registrador indicado.
- NPED ou NPD - (Número pedido) - Requer um dado numérico do meio externo, que será colocado como conteúdo do registrador indicado.

- PARE ou PR - Pára a execução do programa.
- RODE ou RDE - (Roda esquerda) - Idêntica a MOVE, apenas os dígitos de mais alta ordem reaparecem à direita.
- RODD ou RDD - (Roda direita) - Idêntica a MOVD, apenas os dígitos de mais baixa ordem reaparecem à esquerda.
- SOME ou SM - Soma ao registrador indicado o conteúdo da posição de memória indicada. O resultado permanece no registrador.
- SUB ou SB - Subtrai do registrador indicado, o conteúdo da memória indicada. O resultado da subtração permanece no registrador.
- TESTZ ou TTZ- (Testa zero) - Se o conteúdo do registrador indicado for igual a zero, a próxima instrução é pulada.
- TROC ou TRC - Troca os conteúdos dos registradores indicados.
- VA - O contador de instruções vai para a posição de memória indicada.

APÊNDICE A-2

EXEMPLO DE PROGRAMA NA LINGUAGEM 'ASSEMBLER' DO MIC-MAC - Programa para imprimir caracteres.

```

NOME  IMPRIME  15
NOME  PROXIMA  IMPRIME + 3
NOME  ZERO     37
NOME  PULA     38
NOME  DADOS    40

1:  FAÇA  D  4
2:  FAÇA  C  0
3:  LEVE  C  ZERO
4:  FAÇA  C  88
5:  LEVE  C  PULA
6:  FAÇA  A  DADOS
7:  LEVE  A  IMPRIME + 2
8:  FAÇA  A  11
9:  LEVE  A  IMPRIME + 1
10: FAÇA  P  IMPRIME
11: PARE
12: 0
13: 0
14: 0
15: FAÇA  P  + + 3          / IMPRIME
16: 0
17: 0
18: CAR  A  @IMPRIME + 2    / PROXIMA
19: CAR  B  IMPRIME + 2
20: CSEC  B
21: LEVE  B  IMPRIME + 2
22: CAR  C  PULA
23: COMP  A  C
24: VA    35

```

25:	CAR	C	ZERO	60:	47476047
26:	COMP	A	C	61:	47624747
27:	CAR	P	IMPRIME + 1	62:	47474747
28:	FAÇA	B	0	63:	00000088
29:	COMP	B	D	64:	72477247
30:	FAÇA	P	PROXIMA	65:	72474747
31:	RODE	A	2	66:	68474747
32:	CSEC	B		67:	62474747
33:	CIMP	A		68:	47474797
34:	FAÇA	P	29	69:	97474772
35:	CIMP	A		70:	47724772
36:	VA		PROXIMA	71:	47476060
37:	0		/ ZERO	72:	60606047
38:	0		/ PULA	73:	47624747
39:	0			74:	47474747
40:	72474747		/ DADOS	75:	00000088
41:	72474768			76:	72474747
42:	68684747			77:	72474747
43:	62626262			78:	68474747
44:	62474747			79:	62474747
45:	47474772			80:	47474747
46:	72727272			81:	47474772
47:	47476060			82:	47474772
48:	60606047			83:	47476047
49:	47626262			84:	47476047
50:	62624747			85:	~~47624747~~
51:	00000088			86:	47474747
52:	72724772			87:	00000088
53:	72474747			88:	72474747
54:	68474747			89:	72474768
55:	62474747			90:	68684747
56:	47474747			91:	62626262
57:	47474772			92:	62474747
58:	72477272			93:	47474772
59:	47476047			94:	47474772

95: 47476047
 96: 47476047
 97: 47626262
 98: 62624747
 99: 00000000

IMPRESSÃO

M	M	III	CCCCC		M	M	AAAAA	CCCCC		
MM	MM	I	C		MM	MM	A	A	C	
M	M	M	I	C	--	M	M	M	AAAAA	C
M	M	I	C		M	M	A	A	C	
M	M	III	CCCCC		M	M	A	A	CCCCC	

BIBLIOGRAFIA

- 1 - ABELSON, Hal et alii - Logo manual. Cambridge, M.I.T., Artificial Intelligence Laboratory, 1973. Logo memo 7.
- 2 - ALPERT, D. & BITZER, D.L. - Advances in computer based education. Science, 167, May, 1970.
- 3 - CANNARA, Alexander B. - Stanford, Stanford University, Institute for Mathematical Studies in the Social Sciences, Dissertation proposal. January, 1974.
- 4 - _____ & WEYER, Stephen A. - A study of children's programming. Stanford, Stanford University, Institute for Mathematical Studies in the Social Sciences, 1974.
- 5 - CARBONELL, Jaime R. - Artificial Intelligence in CAI: an artificial intelligence approach to CAI. IEEE Transactions on Man-Machine Systems. MMS-11 (4) Dec., 1970.
- 6 - CHU, Yaohan - Introduction to computer organization. Englewood Cliffs, N.J., Prentice-Hall, 1970.
- 7 - GEAR, William C. - Computer organization and programming. New York, McGraw-Hill, 1969.
- 8 - GOLBERG, Adele et alii - What is smart about CAI?. Stanford, Stanford University, Institute for Mathematical Studies in the Social Sciences, 1974.
- 9 - KAY, Alan - A personal computer for children of all ages. In: Annual ACM Conference - Proceedings... New York, ACM, 1972.
- 10 - KOFFMAN, Elliot B. - A generative CAI Tutor for computer science concepts. In: Spring Joint Computer Conference, 1972.
- 11 - KOFFMAN, Elliot B. - An introduction to computer assisted instruction. In: Computing Curriculum Corporation - CAI Curriculum Packages. Palo Alto, 1972.
- 12 - _____ - Design Techniques for generative computer assisted instructional systems. IEEE Transactions on Education. E-16 (4) Nov., 1973.
- 13 - LE FAIVRE, Rick - Implementation of a generative CAI System on a small computer. Madison, University of Wisconsin, Computer

Sciences Department, 1973. WIS-CS-73-177.

- 14 - MINSKY, Marvin - Form and content in computer science. Journal of ACM 17(2) : 197-215, Ap., 1970
- 15 - NIEVERGELT, Jurg et alii - The automation of introductory computer science courses. In: International Computing Symposium- Proceedings... Davos, Switzerland, 1973.
- 16 - NOBLE, H.M. - An author language for computer assisted instruction. Edinburgh, University of Edinburgh, School of Artificial Intelligence, Bionics Research Laboratory, 1973. BRS-10.
- 17 - PAPERT, Seymour - A computer lab for elementary Schools. Computers & Automation 21(6) June, 1972.
- 18 - _____ - On making a theorem for a child. In: Annual ACM Conference - Proceedings... New York, ACM, 1972.
- 19 - _____ - Teaching children thinking. Bulletin of the Association of Teachers of Math (58), 1972.
- 20 - _____ - Teaching children to be mathematicians versus teaching about mathematics. Inter J. Math. Educ. Sci. Technol. 3: 249-262, 1972.
- 21 - _____ - Uses of technology to enhance education. Cambridge Artificial Intelligence Laboratory, M.I.T., Memo n° 298.
- 22 - PAPERT, Seymour & SOLOMON, Cynthia - Twenty things to do with a computer. Educational Technology. Ar., 1972.
- 23 - PDP-8: Laboratory computer Handbook. Digital Equipment Corporation. Maynard, Mass., 1970. Chapter 8.
- 24 - SIMONSEN, Roger H. & RENSHAW, Kent S. - CAI-Boon or Boondoggle? Datamation. March, 1974.
- 25 - SUPPES, Patrick - CAI: an overview of operations and problems. Information Processing 68. Amsterdam, North-Holland, 1969.
- 26 - SUPPES, Patrick - Computer assisted instruction at Stanford. In: Conferencia Internacional sobre o Homem e o Computador, Bordeaux, 1970. Proc. p. 298-330.
- 27 - SEIDEL, Robert J. - Is CAI cost/effective? The right question at the wrong time. Educational Technology, May, 1969.
- 28 - VON STAA, Arndt & LUCENA, Carlos José P. - Simulador PUC 007. Rio de Janeiro, PUC, Departamento de Informática, 1969.

- 29 - WEXLER, Jonathan D. - Information networks in generative CAI. IEEE Transactions on Man - Machine Systems. MMS-11(1) Dec., 1970.,
- 30 - Grupo de Trabalhos Especiais. Consulpuc. Projeto: Sistema PBASICO. Definição preliminar do sistema de ensino; Manual interno. Rio de Janeiro, PUC, Departamento de Informática, 1974. DP21-0005-0.