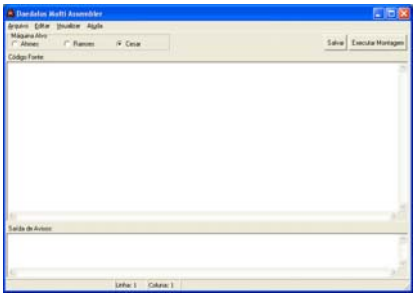
Montador Daedalus

o montador e os simuladores
formatos de instruções

O montador Daedalus

- Desenvolvido para os simuladores Ahmes, Ramses e Cesar
  - simuladores possuem arquiteturas distintas
  - necessário indicar para o Daedalus a arquitetura alvo do código a ser gerado

Interface Daedalus



Programas Daedalus

- Um programa fonte é composto por uma série de linhas de código
- Cada linha contém instruções ou diretivas do montador
- As diretivas são uma versão simplificada das existentes para microprocessadores atuais

Instruções Daedalus

- Uma linha de instrução contém um mnemônico de uma instrução do simulador (Ahmes, Ramses ou Cesar)
- Cada linha gera exatamente uma instrução de máquina
- Formato geral

[Rotulo:] Mnemonico [operandos] [; comentario]

[] : itens opcionais

Campo de rótulo

- Símbolo definido pelo usuário ao qual é atribuído o valor corrente do contador de programa
- Introduzido na tabela de símbolos do montador
- O rótulo deve ser definido uma única vez, e não pode ser redefinido
- Inicia na coluna 1 e deve ser seguido de dois-pontos

[Rotulo:] Mnemonico [operandos] [; comentario]

Mnemônico

- Identifica uma instrução de uma das máquinas (Ahmes, Ramses ou Cesar)
- Corresponde a uma única instrução executável
- Exemplos
  - Ahmes: JNB
  - Ramses: JSR
  - Cesar: MOV

[Rotulo:] Mnemonico [operandos] [; comentario]

Operandos

- Campos de operando podem conter zero ou mais operandos separados conforme a notação utilizada para o código simbólico
- A quantidade de operandos depende da instrução especificada pelo mnemônico
- Os operandos devem ser separados do mnemônico por pelo menos um espaço

[Rotulo:] Mnemonico [operandos] [; comentario]

Linhas contendo diretivas

- Uma linha de diretiva contém comandos para o montador
  - a definição de áreas de variáveis
  - endereços de memória onde a montagem deve ser feita
- Uma diretiva (pseudo-instrução) desenvolve alguma função durante o processo de montagem
- Não produz qualquer código executável
- Pode reservar e inicializar espaço de dados do programa

Formato de Diretivas

- Formato Geral

[Nome:] Diretiva [Operandos] [; Comentario]

- Diretivas implementadas
  - DB
  - DAB
  - DW
  - DAW
  - ORG

Diretiva DB

- Função:
  - reserva um byte para variáveis, permitindo opcionalmente inicializá-lo com um valor decimal, hexadecimal ou ASCII
- Formato:

  [nomeVariavel:] DB valorInicial

Diretiva DW

- Função:
  - reserva dois bytes (uma palavra do processador César = 16 bits) para variáveis, permitindo opcionalmente inicializá-los com um valor decimal, hexadecimal ou ASCII.
- Formato:

  [nomeVariavel:] DW valorInicial

Diretiva DAB

- Função:
  - reserva um array de bytes (8 bits) para variáveis, permitindo opcionalmente inicializá-los valores decimais, hexadecimais ou ASCII
- Pode ser especificado somente um número entre parênteses, indicando a quantidade de bytes a reservar
- Formatos:

```
[nomeVariavel:]    DAB   valor1,valor2,...,valorN
[nomeVariavel:]    DAB   [numeroDeBytes]
```

Diretiva DAW

- Função:
  - reserva um array de palavras (16 bits) para variáveis, permitindo opcionalmente inicializá-las com valores decimais, hexadecimais ou ASCII
- Independente da máquina alvo, será gerado um array usando-se a notação big endian
- Formato:

```
[nome-variavel:]    DAW   valor1,valor2,...,valorN
[nome-variavel:]    DAW   [numeroDePalavras]
```

Diretiva ORG

- Altera o valor do contador de posição
- Posiciona o código gerado a partir do endereço especificado
- É útil para separar a área de códigos da de variáveis
- O montador emite um aviso (Warning) se:
  - sobrescrever uma posição de memória
  - ultrapassar o tamanho de memória existente da máquina específica.
- Formato: ORG posicaoMemoria

Símbolos

- Usado como
  - rótulo de uma instrução
  - nome nas diretivas que geram áreas de dados para referenciar as mesmas nas instruções
- Pode conter até 24 caracteres, de acordo com as seguintes regras:
  - O primeiro caractere não pode ser numérico;
  - Os demais caracteres podem ser a-z, A-Z, 0-9 e o caracter sublinha (_);

Constantes

- Usadas na codificação de instruções e na definição de valores em diretivas
- Uma constante pode ser representada de três maneiras distintas:
  - Decimal
  - Hexadecimal
  - ASCII

Exemplos de diretivas ...

```
Zero:       DB    0       ; define um byte com o valor zero
Um:         DB    1       ; define um byte com o valor um
Menos_um:   DB    -1      ; define um byte com o valor menos um
Quinze:     DB    H0F     ; define um byte com o valor 15 (ou F em hexa)
Letra_A:    DB    'A' ; define um byte com o valor 65 (valor ASCII da letra A)
            DB    'B'     ; define um byte sem nome simbólico
            DB    ' ' ; define um byte com o valor 39 (código do apóstrofe)
            DW    'A'     ; define uma palavra com o primeiro byte em zero
                          ; e o segundo com 65
Palavra0:   DW    0       ; define uma palavra com o valor zero
            DW    H2A1    ; uma palavra com 02 no primeiro byte
                          ; e A1 no segundo
```

Exemplos

```
Vetor:      DAB   10, H15, -5, H0D ; uma seqüência de quatro bytes
Vetor_1:    DAB   [16]; uma seqüência de 16 bytes (inicializados com zero)
            DAB   'A','B','C'; uma seqüência de 3 bytes
            DAB   'ABC'    ; a mesma seqüência anterior, simplificada
            DAW   'ABC'    ; mesma seqüência, armazenada em 3 palavras
            DAW   6500, H0FFF
Msg:        DAB   'Entre com o valor:' ; string com texto
End_Msg:    DW    Msg      ; endereço do string "Msg"
Operando:   DB    0
P_Operando: DB    operando; endereço (de um byte) da variável operando
                  ; use esta forma para os simuladores Ahmes e Ramses
PW_Oper:    DW    operando; endereço (de dois bytes) da variável operando
                  ; use esta segunda forma para o simulador Cesar
```