Departamento de Ciência da Computação – IME-USP

MAC 115 – Introdução à Computação para Ciências Exatas e Tecnologia

ESCOLA POLITÉCNICA – PRIMEIRO SEMESTRE DE 1997

Quarto Exercício-Programa — Data de entrega: até 17 de junho de 1997.

Turmas: Computação; Elétrica A; Elétrica B; Elétrica C; Elétrica D; Mecatrônica.

CALCULADORA

O objetivo deste quarto exercício-programa é construir uma calculadora que trabalhe com números inteiros grandes. O comportamento do seu programa, do ponto de vista de quem usa, deverá ser semelhante a uma calculadora de bolso. Neste EP, você deverá usar as funcões que você construiu no terceiro exercício-programa.

Principais áreas para armazenar os números da calculadora

Para armazenar os valores a serem utilizados nas contas, a calculadora deverá possuir quatro áreas principais: dois `Acumuladores` (que são dois vetores de inteiros), uma `Memória` (um outro vetor de inteiros) e um `Visor` (um string). Cabe aqui observar que se o programa fizer alguma modificação no string `Visor`, este mesmo string deverá ser apresentado na tela do computador.

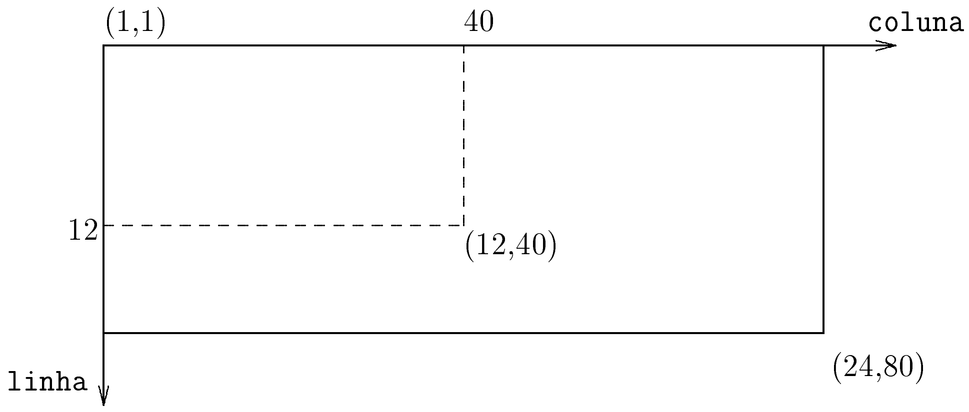
Comandos aceitos pela calculadora

A seguir descreveremos os comandos da calculadora e o seu funcionamento.

- `DES`: Desliga a calculadora, ou seja, encerra o programa.
- `LIM`: Retorna a calculadora ao estado inicial, ou seja, limpa o valor do `Visor`, da `Memória`, dos `Acumuladores` e da operação a ser executada.
- `CE`: Retorna o `Visor` ao valor zero, ou seja, limpa somente o valor do `Visor` sem alterar a última operação, o valor dos `Acumuladores` e o valor da `Memória`.
- `RAIZ`: Calcula uma aproximação inteira da raiz quadrada do número do `Visor`. Por exemplo, se o número do `Visor` é `20`, então este comando devolve o valor `4`.
- `DIV`, `MUL`, `SOM`, `SUB`, `MOD`, `SHOW`: Executa a operação aritmética (`quociente`, `multiplicação`, `soma`, `subtração`, `resto` e `mostra resultado no visor`, respectivamente).
- `ENT`: Este comando serve para fazer uma entrada de um número. Quando este comando for selecionado, o seu programa deverá ler um número do teclado.
- `RVS`: Reverte o sinal do número do `Visor`.
- `RM`: Coloca no `Visor` o número armazenado na `Memória`.
- `M+`: Soma o número armazenado na `Memória` com o `Visor` e armazena o resultado na `Memória`.
- `M-`: Subtrai o número armazenado na `Memória` com o `Visor` e armazena o resultado na `Memória`.
- `CM`: Zera o valor da `Memória`.

Representação da calculadora na tela

A tela pode ser vista como uma matriz (linha, coluna), onde o canto superior esquerdo tem a posição (1,1) e o canto inferior direito tem posição (24,80).



Algumas funções para tela

(*a*) Função _settextposition (lin, col)

Para mover o cursor na tela para a posição (lin, col), use a função

_settextposition (lin, col)[1];

Por exemplo, _settextposition (12,40); coloca o cursor no ponto (12,40) da tela.

Para quem estiver utilizando o compilador TurboC, a função que move o cursor é

gotoxy (col, lin)[2];

Por exemplo, gotoxy (40, 12); coloca o cursor no ponto (12, 40) da tela. Note que há uma inversão entre lin e col.

(*b*) Função _clearscreen (_GCLEARSCREEN)

Para "apagar" a tela e colocar o cursor na posição (1,1) utilize a função

_clearscreen (_GCLEARSCREEN);

Para quem estiver utilizando o compilador TurboC, utilize a função

clrscr();

[1] Você deve colocar # include <graph.h> no início de seu programa
[2] Você deve colocar # include <conio.h> no início de seu programa

Bordas da calculadora

Para as bordas da calculadora utilize os seguintes caracteres:

caractere	código ASCII
│	179
┐	191
└	192
─	196
┘	217
┌	218

Entrada e saída do seu programa

As teclas da calculadora podem ser selecionadas pelo usuário do programa como opções de um "menu" que você pode mostrar permanentemente na tela do vídeo[3].

```
                     PROGRAMA CALCULADORA

   VISOR:                                              0

   01.  DES     02.  LIM     03.  CE      04.  RVS

   05.  RM      06.  CM      07.  M+      08.  M-

   09.  RAIZ    10.  SOM     11.  SUB     12.  MUL

   13.  DIV     14.  MOD     15.  SHOW    16.  ENT

   Opcao:  __
```

A calculadora deverá permitir a entrada dos números e operações em qualquer ordem, fornecendo no visor os valores parciais dos resultados das operações, como as calculadoras fazem usualmente. Esse visor deve ainda informar o seguinte:

(a) Se algum valor estiver na `Memória`, deve ser informado no visor através de um `M` (0 não é considerado valor).

(b) Se algum erro ocorrer (divisão por zero, ultrapassar a capacidade, entrada não é número, raiz de número negativo, etc.) isso deve ser informado no visor através de uma mensagem conveniente.

A implementação da calculadora deverá cuidar de alguns detalhes:

(1) O número máximo de dígitos deve ser fixado através de uma constante (60, por exemplo). Para isso, como no terceiro EP, coloque no início do seu programa:

```
# define MAX_DIGITOS 60
```

[3] se você quiser sofisticar, fique à vontade

(2) Após a escolha do comando ENT, a entrada de um número inteiro deverá ser feita através de string (para isso utilize a função pega_string do terceiro EP). Não é necessário ir atualizando o visor enquanto o usuário ainda estiver digitando o número.

(3) Se duas teclas de operação (MUL, DIV, etc.) forem digitadas, considere apenas a última (como faz a maioria das calculadoras).

(4) A calculadora deverá ter precedência de operadores. Por exemplo, 3+3*5 deve dar como resultado 18 e não 30. A tabela de precedência em ordem decrescente é:

MUL, DIV, MOD
SOM, SUB

Para isso, você deve usar adequadamente os dois Acumuladores!!!

Outros exemplos: 3+14/5=3+2=5, 4-13%3=4-1=3, 14/3/2=4/2=2
3+2*3*4*5*6+7=3+720+7=723+7=730.

Faça também uma saída para arquivo, de forma que você possa seguir todas opções que foram realizadas pelo seu programa quando executado. Isto funcionará como um pequeno relatório gerado pelo seu programa e facilitará a correção do mesmo (você deverá imprimi-lo para entregá-lo juntamente com um disquete e uma listagem de seu programa).

Funções que você deve implementar neste EP

Neste exercício-programa, você deve construir **obrigatoriamente** as funções descritas a seguir.

1. `int operacao_soma (int num1[], int num2[], int res[]);`

   Observe que se o usuário selecionar a operação SOM, o seu programa deverá ter cuidado com os sinais dos números. Por exemplo,

   $$-3 + 2 = -1 = 2 - 3.$$

   $$4 + -2 = 2 = 4 - 2.$$

   As operações que são somas, na verdade são subtrações.

   Esta função deve analisar os sinais dos números num1 e num2 e chamar a função soma ou subtrai do terceiro EP, e colocar o resultado correto em res. No caso de ocorrer overflow, esta função deve retornar valor um. Caso contrário, a função deve retornar valor zero.

2. `int operacao_subtracao (int num1[], int num2[], int res[]);`

   O mesmo acontece se o usuário escolher a operação SUB. Observe os seguintes exemplos:

   $$-4 - 2 = -6 = -(4 + 2)$$

   $$-5 - -2 = -3 = -5 + 2$$

   As operações que eram subtrações viraram somas.

Esta função deve analisar os sinais dos números `num1` e `num2` e chamar a função `soma` ou `subtrai` do terceiro EP, e colocar o resultado correto em `res`. No caso de ocorrer overflow, esta função deve retornar valor um. Caso contrário, a função deve retornar valor zero.

3. `int operacao_quociente_resto (int num1[], int num2[], int quoc[], int rest[]);`

   Suponha que você queira dividir 4078 por 38. Você conhece um algoritmo usual de fazer essa divisão:

$$
\begin{array}{r|l}
4078 & 38 \\ \cline{2-2}
38 & 107 \\ \cline{1-1}
278 & \\
266 & \\ \cline{1-1}
12 &
\end{array}
$$

   Esta função, usando **obrigatoriamente** este algoritmo, deve fazer a divisão de `num1` por `num2` e encontrar o quociente (que deve ser armazenado em `quoc`) e o resto (que deve ser guardado em `rest`).

   Para isso, siga os seguintes passos:

   Passo 1: Encontre o subnúmero de `num1`, da esquerda para a direita, que é imediatamente maior que `num2`. No exemplo anterior, esse número é `40`.

   Passo 2: Encontre o quociente e o resto da divisão desse número por `num2` usando as funções `quociente` e `resto` do terceiro EP. No exemplo, os resultados são `1` e `2`, respectivamente. Coloque adequadamente o quociente obtido em `quoc`.

   Passo 3: A partir do resto obtido no passo anterior, construa um número colocando os próximos algarismos de `num1` até que ele seja imediantemente superior a `num2`, lembrando, se for o caso, de acrescentar zeros em `quoc`.

   Passo 4: Repita o Passo 2 e 3, até terminar a divisão.

   Passo 5: Coloque o último resto obtido em `rest`.

   Observações:

   (*a*) No caso do vetor `num2` ser zero, a função deve retornar valor 2 (dois). Caso contrário, a função deve retornar valor zero.

   (*b*) **Sinal do resto:**

   Calcula-se o resto considerando `num1` e `num2` positivos. Após este cálculo, o sinal do resto é o mesmo sinal de `num1`.

4. `int operacao_raiz (int numero[], int res[]);`

   Calcula uma aproximação inteira da raiz quadrada do vetor `numero` e armazena o resultado no vetor `res`. No caso do vetor `numero` ser negativo, a função deve retornar valor 4 (quatro). Caso contrário, a função deve retornar valor zero.

Modificar algumas funções do terceiro EP

Neste EP você deve modificar algumas funções do exercício-programa anterior.

1. `void trata_erro (int tipo_do_erro);`

   Nesta função, acrescente mais uma situação de erro: raiz quadrada de um número negativo. Considere agora os seguintes códigos de erro:

`tipo_do_erro`	erro
01	overflow
02	divisão por zero
03	entrada não é número
04	raiz de número negativo

Faça pelo menos este teste!

Faça quantos testes você achar necessário, mas execute pelo menos estes que se seguem e entregue-os juntamente com a listagem do seu programa.

(a) 999999999999999999999999999999999999999999999999999999999999 + 3.

(b) 123456789123456789 + 123456789.

(c) 123456789123456789 − 987654321.

(d) 98765432 − 123456789.

(e) 1234567 × 987654321.

(f) −23456789 × 987654.

(g) 987654321987654321987654321987654321987654321987654321 × 987654321.

(h) 987654321 / 123456789.

(i) 87654321 / −23456789.

(j) 12345678 / 0.

(k) 987654321 mod 123456789.

(l) 98765432 mod 0.

(m) reverta o sinal de 987654.

(n) tente ler como número o string 987po1209sfa