MAC 110 — Introdução à Computação

BCC — Primeiro Semestre de 2007

Segundo Exercício-Programa

Prazo de entrega: até **15 de maio de 2007.**

Raízes de Equações Quadráticas

Escreva um programa em C que calcula as raízes de equações quadráticas. Seu programa deve ler um real $\varepsilon > 0$ que define a precisão para cálculo de raízes quadradas, um inteiro $n \geq 1$ e os coeficientes reais a, b e c de n equações do segundo grau ($ax^2 + bx + c = 0$, com $a \neq 0$). O programa deve calcular as raízes de cada equação e imprimir os resultados da maneira especificada abaixo.

Impressão dos resultados

Para cada equação deve-se imprimir seus coeficientes, o tipo de suas raízes (reais simples, real dupla ou complexas) e as raízes. No caso em que as raízes forem complexas, deve-se imprimir a parte real e a parte imaginária. Exemplo: Para a entrada da Figura 1, a saída do programa deve ser como a apresentada na Figura 2.

```
1.0e-3
8
1      -2      -3
1      -2      10
1      -2       1
7.18   43.75  -31.21
3     -18      27
0      10      20
1       2       3
1       0      -2
```

Figura 1: Exemplo de entrada para o programa de cálculo de raízes de equações quadráticas

```
                     Raízes de Equações Quadráticas
--------------------------------------------------------------------------------
     coeficientes        tipo das
   a      b      c       soluções       raiz 1                raiz 2
--------------------------------------------------------------------------------
  1.00  -2.00  -3.00   reais simples   3.0000               -1.0000
  1.00  -2.00  10.00   complexas       1.0000 + i*3.0000     1.0000 - i*3.0000
  1.00  -2.00   1.00   real dupla      1.0000                1.0000
  7.18  43.75 -31.21   reais simples   0.6451               -6.7384
  3.00 -18.00  27.00   real dupla      3.0000                3.0000
  0.00  10.00  20.00   *** ERRO: equação não é do segundo grau! ***
  1.00   2.00   3.00   complexas      -1.0000 + i*1.4142    -1.0000 - i*1.4142
  1.00   0.00  -2.00   reais simples   1.4142               -1.4142
```

Figura 2: Saída correspondente à entrada na Figura 1

Cálculo da raiz quadrada de um número real não negativo

Neste exercício você não usará a função `sqrt(x)` da biblioteca `math.h`. Seu programa deve ter obrigatoriamente uma função com protótipo

```
float raiz_quadrada(float x, float epsilon);
```

que calcula a raiz quadrada de `x` usando o método de Newton, descrito a seguir.

Suponha que desejamos extrair a raiz quadrada de um número real $x > 0$. Escolhe-se como chute inicial para $\sqrt{x}$ o número $r_0 = x$ e calcula-se a seguinte seqüência de números:

$$r_{n+1} = \frac{1}{2}\left(r_n + \frac{x}{r_n}\right) \qquad n = 0, 1, 2, \ldots$$

(Ou seja: obtemos

$$r_1 = \frac{1}{2}\left(r_0 + \frac{x}{r_0}\right) = \frac{1}{2}\left(x + \frac{x}{x}\right) = \frac{x+1}{2},$$

a partir de r_1 obtemos r_2 e assim por diante.)

Esse processo deve ser repetido enquanto $|r_{n+1} - r_n| \geq \varepsilon$, onde ε é um número positivo dado que representa a precisão do cálculo da raiz. A aproximação de $\sqrt{x}$ será o primeiro valor r_{n+1} para o qual $|r_{n+1} - r_n| < \varepsilon$. Utilize como ε o valor do parâmetro `epsilon` passado à função `raiz_quadrada`.

Os tipos `double` e `long double`

Vimos em classe apenas um tipo de dados para números "reais": `float`, que é uma representação em ponto flutuante com precisão simples. A linguagem C tem outros dois tipos "reais" que podem ser usados quando se precisa de maior precisão: `double` (precisão dupla) e `long double` (precisão extendida). Exemplos de utilização:

```
/* declarações de variáveis float, double e long double: */
float f;
double d;
long double ld;

/* constantes float, double e long double: */
f = 3.14f;
d = 3.14;   /* ausência de sufixo "f" ou "L" indica double */
ld = 3.14L;

/* leitura de float, double e long double: */
scanf("%f %lf %Lf", &f, &d, &ld);

/* impressão de float, double e long double: */
printf("%f %f %Lf", f, d, ld);
```

Note que a leitura e a impressão de `double`s são feitas com especificações de formato diferentes: a leitura é com `"%lf"` e a impressão é com `"%f"`. Repare também que a mesma especificação de formato (`"%f"`) é usada para imprimir tanto `float`s como `double`s. (A impressão com `"%f"` é, na verdade, sempre de `double`s. Quando se imprime um `float` com `"%f"`, o `float` é automaticamente "promovido" para `double` antes de ser passado como parâmetro à função `printf`.)

Faz diferença usar `float`, `double` ou `long double`?

Para verificar a influência do tipo "real" utilizado, faça os experimentos descritos abaixo. Use inicialmente uma versão do seu programa na qual todas as variáveis "reais" são do tipo `float`.

1. Rode o seu programa com valores cada vez menores do parâmetro ε. Começe com $\varepsilon = 10^{-3}$, como no arquivo de entrada da figura 1, e depois tente usar potências de 10 cada vez menores: $\varepsilon = 10^{-3}, 10^{-4}, 10^{-5}, \ldots$ Até que precisão o programa funciona? O que acontece quando o programa deixa de funcionar?

2. Crie uma nova versão do seu programa que usa precisão dupla (`double`) para calcular raízes quadradas. Nesta versão do programa, a função `raiz_quadrada` terá protótipo

   ```
   float raiz_quadrada(double x, double epsilon);
   ```

   Embora essa versão da função devolva a raiz quadrada como um `float`, internamente ela trabalha só com `double`s. Você pode continuar usando `float`s no restante do programa, exceto quando estiver lendo o valor de ε da entrada. Esse valor deve ser lido como `double`.

3. Repita o ítem 1 com a segunda versão do programa. Até que precisão essa versão do programa funciona?

4. Crie uma terceira versão do seu programa que usa precisão extendida (`long double`) para calcular raízes quadradas. Nesta versão do programa, a função `raiz_quadrada` terá protótipo

   ```
   float raiz_quadrada(long double x, long double epsilon);
   ```

   Embora essa versão da função devolva a raiz quadrada como um `float`, internamente ela trabalha só com `long double`s. Você pode continuar usando `float`s no restante do programa, exceto quando estiver lendo o valor de ε da entrada. Esse valor deve ser lido como `long double`.

5. Repita o ítem 1 com a terceira versão do programa e descubra até que precisão essa versão do programa funciona.

Que versão do programa deve ser entregue?

Entregue apenas a versão que usa precisão dupla (`double`) para calcular raízes quadradas, mas coloque num comentário no início do programa, logo após o cabeçalho com seu nome e número USP, suas respostas para as seguintes questões:

(a) Até que precisão funcionou o programa só com `float`s?

(b) Até que precisão funcionou o programa que usa precisão dupla para calcular a raízes quadradas?

(c) Até que precisão funcionou o programa que usa precisão extendida para calcular a raízes quadradas?

(d) Para cada uma das três versões do programa, explique também o que acontece quando a precisão é excessiva e o programa deixa de funcionar.

Observações Importantes

1) Todos os exercícios-programa devem ter um cabeçalho como o seguinte:

```
/*******************************************************/
/* Aluno: Fulano de Tal                                */
/* Número USP: 12345678                                */
/* Exercicio-Programa 2 -- Raízes de Equações Quadráticas */
/* MAC110 (BCC) -- 2007 -- Professor: Reverbel         */
/* Compilador: ... (DevC++ ou gcc) versão ...          */
/*******************************************************/
```

2) O exercício-programa é estritamente individual. Exercícios copiados (com ou sem eventuais disfarces) receberão nota ZERO.

3) Exercícios atrasados não serão aceitos.

4) Exercícios com erros de sintaxe (ou seja, erros de compilação) receberão nota ZERO.

5) É muito importante que seu programa tenha comentários e esteja bem indentado, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa (conforme visto em aula). A avaliação dos exercícios-programa levará isto em conta.

6) Cada programa deve ter sido executado tantas vezes quantas forem necessárias para testar os vários casos possíveis para as entradas.

7) Você entregará seu exercício-programa através do sistema Paca/Moodle (`http://paca.ime.usp.br`). Para isto você precisa estar cadastrado nesse sistema (use o seu número USP para se cadastrar) e registrado no Paca como aluno da disciplina `MAC110-BCC` (uma vez cadastrado no Paca, basta entrar na área da disciplina `MAC110-BCC` que o sistema perguntará a você se deseja se registrar como aluno da disciplina).

8) Entregue apenas o programa fonte em C, num arquivo com nome `ep2-`<*seu-número-USP*>`.c`. (Exemplo: se seu número USP for 12345678, você deverá entregar um arquivo `ep2-12345678.c`.)

9) Enquanto o prazo de entrega não expirar, você poderá entregar várias versões do mesmo exercício-programa. Apenas a última versão entregue será guardada pelo sistema. Encerrado o prazo, o sistema não aceitará mais a entrega de exercícios-programa. Não deixe para entregar seu exercício na última hora!

10) Guarde uma cópia do seu exercício-programa pelo menos até o final do semestre.

Bom trabalho!