MAC2166 – Introdução à Computação para Engenharia

Escola Politécnica

Segunda Prova – 17 de maio de 2010

Nome: _____

Assinatura: _____

Nº USP: _____ Turma: _____

Professor: _____

Instruções:

1. Não destaque as folhas deste caderno.
2. A prova consta de 4 questões. Verifique antes de começar a prova se o seu caderno de questões está completo.
3. A prova pode ser feita a lápis. Cuidado com a legibilidade.
4. Qualquer questão pode ser resolvida em qualquer página. Se a questão não está na página correspondente ao enunciado basta indicar isto na página e escrever QUESTÃO X em letras ENORMES antes da solução.
5. Não é necessário apagar rascunhos no caderno de questões.
6. Não é permitido o uso de folhas avulsas para rascunho.
7. Não é permitido o uso de calculadoras.
8. Não é permitido a consulta a livros, apontamentos ou colegas.

DURAÇÃO DA PROVA: 2 horas

Questão	Valor	Nota
1	2.5	
2	2.5	
3	2.5	
4	2.5	
Total	10.0	

Questão 1

Simule a execução do programa abaixo, destacando a sua saída. A saída do programa consiste de tudo que resulta dos comandos `printf`. **Para efeito de correção só será considerada a saída do programa.**

```
# include <stdio.h>

int f1 (int a, int *b) {
  int z;
  *b = *b - a;
  a = a + *b;
  z = a - *b;
  return z;
}
int f2 (int *a, int b) {
  int z;
  *a = *a - b;
   b = *a + b;
   z = b - *a;
  return z;
}
int f3 (int *a, int b) {
  b = *a + b;
  *a = b - *a;
  return b;
}

int main () {
  int nusp;
  int a, b, c, d, e;
  float f;
  printf ("Entre com seu no. USP: ");
  scanf ("%d", &nusp); /* use aqui seu numero USP */
  printf ("nusp = %d\n", nusp);
  a = (nusp % 10) + 1;
  b = a % 3;
  printf ("1: a=%d b=%d\n", a, b);
  c = a;
  d = b;
  e = f1 (c, &d);
  printf ("2: a=%d b=%d c=%d d=%d e=%d\n", a, b, c, d, e);
  d = b;
  e = f2 ( &a, d);
  printf ("3: a=%d b=%d c=%d d=%d e=%d\n", a, b, c, d, e);
  a = f3 ( &c, c);
  printf ("4: a=%d b=%d c=%d\n", a, b, c);
  f = d / (b + 1);
  printf ("5: a=%d b=%d d=%d f=%f\n", a, b + 1, d, f);
  f = ((float) (2 * a + 1)) / 2;
  e = f;
  printf ("6: a=%d b=%d e=%d f=%f\n", a, b, e, f);
  return 0;
}
```

Dados para a simulação: **o seu número USP**

main							
a	b	c	d	e	f	g	nusp

f1		
a	b	z

f2		
a	b	z

f3	
a	b

saída

```
Entre com o seu no. USP:
```

Questão 2

Considere o seguinte problema: *Dada uma sequência de caracteres terminada por um ponto '.', representando um texto, determinar os tamanhos da palavra mais curta (menor) e da palavra mais longa (maior) do texto.* Para isso, considere uma palavra como uma sequência construída exclusivamente por letras, terminada quando o caractere seguinte não é letra. Por exemplo, para o texto (da poesia *Filosofia*-Ascenso Ferreira):

```
Hora de trabalhar? - Pernas pro ar que ninguem eh de ferro!.

Maior Palavra: 9
Menor Palavra: 2
```

Para auxiliar na solução deste problema foi implementada uma função `contaletras` que lê uma seqüência de caracteres até terminar uma palavra, retorna o último caractere lido (que não está na palavra lida) e ainda devolve num parâmetro quantos caracteres foram lidos. Veja a implementação da `contaletras`:

```
#include <stdio.h>
char contaletras (int *tam){
 char c = 'a';
 *tam = 0;
 while((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) {
    scanf( "%c", &c );
    *tam = *tam + 1;
 }
 return c;
}
```

Quatro possíveis soluções para o problema estão apresentadas na página ao lado. Cada "solução" é um programa que lê um texto terminado por '.' e imprime os tamanhos da maior e da menor palavra. Alguns testes foram executados para este problema. Para cada uma das saídas seguintes, assinale **UM** dos programas que poderiam ter sido utilizados para produzir tais resultados. Note que mais de um programa poderá ter dado os mesmos resultados.

saída do programa	**assinale um dos possíveis programas utilizado**
Digite um texto terminado por . : casa. <não termina>	☐ prog1 ☐ prog2 ☐ prog3 ☐ prog4
Digite um texto terminado por . : casa. Maior Palavra = 4 Menor Palavra = 4	☐ prog1 ☐ prog2 ☐ prog3 ☐ prog4
Digite um texto terminado por . : casa bonita!. Maior Palavra = 7 Menor Palavra = 1	☐ prog1 ☐ prog2 ☐ prog3 ☐ prog4
Digite um texto terminado por . : casa bonita!. <não termina>	☐ prog1 ☐ prog2 ☐ prog3 ☐ prog4
Digite um texto terminado por .: casa bonita!. Maior Palavra = 6 Menor Palavra = 4	☐ prog1 ☐ prog2 ☐ prog3 ☐ prog4

```
/********
* prog1 *
********/

int main() {
  char c; /* caracter   */
  int tam_palavra = 0, tam_maior = 0, tam_menor = 0;

  printf("\nDigite um texto terminado por . : \n\n");
  scanf("%c",&c);
  while(c != '.'){
     c = contaletras (&tam_palavra);
     if(tam_palavra > tam_maior)
            tam_maior = tam_palavra;
     if(tam_palavra < tam_menor || tam_menor == 0)
            tam_menor = tam_palavra;
  }

  printf("\n Maior Palavra = %d \n Menor Palavra = %d", tam_maior, tam_menor);
  return 0;
}
```

```
/********
* prog2 *
********/

int main() {
  char c; /* caracter   */
  int tam_palavra = 0, tam_maior = 0, tam_menor = 0;

  printf("\nDigite um texto terminado por . : \n\n");
  scanf("%c",&c);
  while(c != '.'){
     while (!(c >= 'A' && c <= 'Z')   ||   !(c >= 'a' && c <= 'z'))
       scanf("%c",&c);
     c = contaletras (&tam_palavra);
     if(tam_palavra > tam_maior)
            tam_maior = tam_palavra;
     if(tam_palavra < tam_menor || tam_menor == 0)
            tam_menor = tam_palavra;
  }

  printf("\n Maior Palavra = %d \n Menor Palavra = %d", tam_maior, tam_menor);
  return 0;
}
```

```
/********
* prog3 *
********/

int main() {
  char c; /* caracter   */
  int tam_palavra = 0, tam_maior = 0, tam_menor = 0;

  printf("\nDigite um texto terminado por .: \n\n");
  scanf("%c",&c);
  while(c != '.'){
     while (!(c >= 'A' && c <= 'Z')  &&  !(c >= 'a' && c <= 'z')  &&  (c != '.'))
        scanf("%c",&c);
     c = contaletras (&tam_palavra);
     if(tam_palavra > tam_maior)
            tam_maior = tam_palavra;
     if(tam_palavra < tam_menor || tam_menor == 0)
            tam_menor = tam_palavra;
  }

  printf("\n Maior Palavra = %d \n Menor Palavra = %d", tam_maior, tam_menor);
  return 0;
}
```

```
/********
* prog4 *
********/

int main() {
  char c; /* caracter   */
  int tam_palavra = 0, tam_maior = 0, tam_menor = 0;

  printf("\nDigite um texto terminado por . : \n\n");
  scanf("%c",&c);
  while(c != '.'){
     if (!(c >= 'A' && c <= 'Z')  &&  !(c >= 'a' && c <= 'z'))
          scanf("%c",&c);
     else{
          c = contaletras (&tam_palavra);
          if(tam_palavra > tam_maior)
               tam_maior = tam_palavra;
          if(tam_palavra < tam_menor || tam_menor == 0)
               tam_menor = tam_palavra;
     }
  }

  printf("\n Maior Palavra = %d \n Menor Palavra = %d", tam_maior, tam_menor);
  return 0;
}
```

Questão 3

Dada uma sequência $x_1, x_2, \ldots, x_k$ de números, suas *permutações cíclicas* são as sequências obtidas sucessivamente mandando o primeiro elemento para o fim. Por exemplo, as permutações cíclicas de $(3, 2, 4)$ são: $(2, 4, 3)$, $(4, 3, 2)$ e $(3, 2, 4)$. Em geral, se a sequência tem k termos, ela tem k permutações cíclicas.

Dizemos que uma seqûencia (a, b, c, d, e) é do *tipo* P2Q3 se a está no intervalo fechado

$$\left[(b-c)^7 - (c^6 - d^5)^6 + d^8 - e^4 \quad , \quad (b-c)^7 + (c^6 - d^5)^6 + d^8 + e^4\right].$$

O problema é escrever um programa que leia uma sequência de 5 números reais, e imprima:

- `SIM`, se existe alguma permutação cíclica dessa sequência que é do tipo P2Q3,
- `NAO`, se nenhuma permutação cíclica dessa sequência é do tipo P2Q3.

Obs.: Você pode supor que o programa vai rodar num computador com capacidade ilimitada; ou seja, não tem perigo de dar estouro ao fazer as contas.

Questão 4

Esta questão consiste em implementar uma função baseada no EP3. Para isso, você **deve** usar as funções a seguir **sem escrevê-las**. Suponha que lhe é dada uma função de protótipo

```
float distancia(float x1, float y1, float x2, float y2);
```

que recebe as coordenadas cartesianas de dois pontos (`x1`, `y1`) e (`x2`, `y2`) e retorna a distância entre eles. Suponha ainda que lhe é dada uma função de protótipo

```
void aceleracao_resultante(float x, float y, float *ax, float *ay);
```

que recebe as coordenadas (`x`, `y`) de uma nave e devolve em (`*ax`, `*ay`) o vetor aceleração da força gravitacional resultante exercida **pela Terra e pela Lua** sobre a nave.

Um corpo celeste famoso por sua inexistência é a **chaleira de Russell** (*Russell's teapot*) que está em órbita circular uniforme em torno da Terra a uma distância `R` do seu centro $(0, 0)$, dando uma volta a cada `P` unidades de tempo. Digamos que a chaleira de Russell começa a sua órbita a partir da posição `(0,R)` no instante $\texttt{t}_0 = 0$ e no instante `t` suas coordenadas são $(\texttt{R}\cos\frac{2\pi}{\texttt{P}}\texttt{t}, \texttt{R}\,\text{sen}\,\frac{2\pi}{\texttt{P}}\texttt{t})$.

Você deve escrever uma função de protótipo

```
int simule(float x, float y, float vx, float vy, float R, float P,
           float deltaT, float maxT, float distmin);
```

que **recebe**

- a posição `(x,y)` e velocidade `(vx,vy)` de uma nave no instante inicial $\texttt{t}_0 = 0$;
- o raio `R` e o período de tempo `P` da órbita da chaleira de Russell em torno da Terra;
- um intervalo de tempo `deltaT`, um tempo máximo de simulação `maxT` e uma distância `distmin`.

A função deve calcular a trajetória da nave, sujeita às forças gravitacionais da Terra e da Lua, começando na posição inicial `(x,y)` com velocidade `(vx,vy)`. O instante inicial da simulação é 0 e o instante seguinte a um dado instante `t` é o instante `t + deltaT`. A simulação da trajetória deverá terminar assim que a nave colidir com a chaleira de Russel ou assim que o tempo de simulação ultrapassar `maxT` ou assim que a nave cruzar a órbita da chaleira (isto é, ficar a uma distância maior que `R` do centro da Terra). Dizemos que a nave *colide* com a chaleira de Russell se a distância entre elas é menor que `distmin`. A função deve **retornar** 1 se a nave colidir com a chaleira de Russel, em caso contrário a função deve **retornar** 0.

Lembre que dados a posição x, a velocidade v e a aceleração a num instante t, podemos calcular seus valores x', v' no instante seguinte $t + \Delta t$ pelas fórmulas:

$$\begin{aligned} x' &= x + v\Delta t \ , \\ v' &= v + a\Delta t \ . \end{aligned}$$

O uso da biblioteca matemática é livre. Aqui vai um pequeno extrato da `math.h`:

```
#define PI 3.1415
double sin(double x);
double cos(double x);
double tan(double x);
double asin(double x);
double acos(double x);
double atan(double x);
```