PROVA DE PROGRAMAÇÃO DE COMPUTADORES I

QUARTA UNIDADE (PP4)

CONTEÚDO: RECURSIVIDADE E ARQUIVOS

Sistemas de Informação Ano 2019 – UEMS

Professora: Mercedes Gonzales Márquez

Data de Entrega: Até 19 horas do 18/11/2019.

Enviar no email mercedes@comp.uems.br com Assunto: Prova PP4 de Programação.

Data da Avaliação Oral PO4: 18/11/2019

RECURSIVIDADE

Fatoriais

1. O superfatorial de um número N é definido pelo produto dos N primeiros fatoriais de N. Assim, o superfatorial de 4 é
   sf(4) = 1! * 2! * 3! * 4! = 288
   Faça uma função recursiva que receba um número inteiro positivo N e retorne o superfatorial desse número.
2. O hiperfatorial de um número N, escrito H(n), é definido por:

$$H(n) = \prod_{k=1}^{n} k^k = 1^1 \cdot 2^2 \cdot 3^3 \cdots (n-1)^{n-1} \cdot n^n$$

   Faça uma função recursiva que receba um número inteiro positivo N e retorne o hiperfatorial desse número.
3. Um fatorial exponencial é um inteiro positivo N elevado à potência de N-1, que por sua vez é elevado à potência de N-2 e assim em diante. Ou seja,

$$n^{(n-1)^{(n-2)^{\cdots}}}$$

   Faça uma função recursiva que receba um número inteiro positivo N e retorne o fatorial exponencial desse número.

Números Tribonacci e Tetranacci

4. Os números tribonacci são definidos pela seguinte recursão.

$$f(n) = \begin{cases} 0 & se\ n = 0 \\ 0 & se\ n = 1 \\ 1 & se\ n = 2 \\ f(n-1) + f(n-2)) + f(n-3) & se\ n > 2 \end{cases}$$

   Faça uma função recursiva que receba um número N e retorne o N-ésimo termo da sequência de tribonacci.

5. Os números tetranacci iniciam com quatro termos pré-determinados e a partir daí todos os demais números são obtidos pela soma dos quatro números anteriores. Os primeiros números tetranacci são: 0, 0, 0, 1, 1, 2, 4, 8, 15, 29, 56, 108, 208...

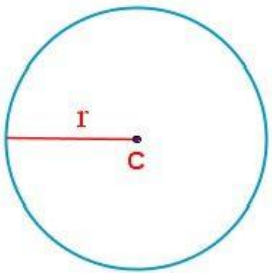
   Faça uma função recursiva que receba um número N e retorne o N-ésimo termo da sequência de tetranacci.

ARQUIVOS

6. Um segmento de reta S é uma porção de uma reta que se encontra entre dois pontos. Assumindo a estrutura Ponto2d vista em aula, a estrutura para o segmento de reta seria a seguinte:

```
typedef struct {
      Ponto2d p1,p2;          /* pontos extremos do segmento de reta */
} segmento;
```

   Já uma circunferência é formada por um conjunto de pontos que são equidistantes ao centro C desta circunferência. Essa distância é chamada de raio r da circunferência. Assim, uma circunferência é naturalmente representada pelo seu centro e raio.



   Assumindo a estrutura Ponto2d vista em aula, a estrutura para uma circunferência seria a seguinte:

```
typedef struct {
      Ponto2d c;     /* centro da circunferência */
      float r;       /* raio da circunferencia */
} circunferencia;
```

   Polígonos são cadeias fechadas de segmentos de reta que não se cruzam. Em lugar de explicitamente listar os segmentos ou arestas do polígono, nós podemos implicitamente representá-lo listando os n vértices que formam a borda do polígono. Assim um segmento existe entre o i-ésimo e o (i+1)-ésimo pontos na cadeia para 0<=i<=n-1.
   Assumindo a estrutura Ponto2d vista em aula, a estrutura para um polígono seria a seguinte:

```
typedef struct {
      int n;              /* numero de pontos do poligono */
      Ponto2d p[MAXPOLY];     /* array de pontos no polygono */
} poligono;
```

   Vamos considerar que queremos salvar as informações da lista de elementos geométricos: segmentos de reta, circunferências e polígonos. A lista seria da seguinte forma:

```
SEGMENTOS DE RETA
P1,P2

CIRCULOS
c, r

POLIGONOS
n
P1
P2
 …
Pn
```

Faça um programa com as seguintes opções:

(1) a criação de um arquivo elementos.txt com diversos elementos geométricos dos três tipos apresentados ingressados pelo teclado.

(2) a criação de um arquivo areas.txt que contenha a área das circunferências e polígonos contidos no arquivo elementos.txt. Verifique a fórmula da área de um polígono na prova anterior PP3.

(3) A inserção de novos elementos geométricos no arquivo já existente elementos.txt.