1ª Prova de Algoritmos e Estruturas de Dados I - Curso: Engenharia de Computação

Data: 13 de setembro de 2018 **Horário limite:** 15:40

Perguntas comuns e suas respostas:

- P: Tenho uma dúvida na questão tal.
  R: A compreensão do enunciado faz parte da prova.
- P: O que será corrigido?
  R: A lógica, a criatividade, a sintaxe, o uso correto dos comandos e dos tipos, os nomes das variáveis, a indentação, uso equilibrado de comentários no código e, evidentemente, a clareza. Nesta prova, você deverá sobretudo escrever códigos modulares, usando corretamente funções e/ou procedimentos, conforme o caso, além de uso correto de variáveis locais ou globais e a passagem de parâmetros por referência ou por valor.
- P: Posso fazer a lápis?
  R: Não. A prova deverá ser feita a caneta.
- P: Posso responder na folha de questões?
  R: Não. A prova deverá ser respondida na folha de respostas.

(5 pontos) 1) Qual o uso da função `scanf`? Descreva o que representa o valor de retorno dessa função.

(10 pontos) 2) Na linguagem C temos a função `malloc()`, fornecida pela `stdlib.h`. Qual o uso desta função? O que a função retorna? Mostre um exemplo de uso desta função e explique todos os passos.

(5 pontos) 3) Diferencie passagem de parâmetro por referência e por cópia. Mostre um exemplo de cada.

(04 pontos) 4) Seja *vet* um vetor de 4 elementos: $TIPOvet[4]$. Supor que depois da declaração, vet esteja armazenado no endereço de memória 4092 (ou seja, o endereço de $vet[0]$). Supor também que na máquina usada uma variável do tipo char ocupa 1 byte, do tipo int ocupa 2 bytes, do tipo float ocupa 4 bytes e do tipo double ocupa 8 bytes.

Qual o valor de $vet + 1$, $vet + 2$ e $vet + 3$, se:

a) *vet* for declarado como *char*?

b) *vet* for declarado como *int*?

c) *vet* for declarado como $float$?

d) *vet* for declarado como *double*?

Justifique a sua resposta.

(16 pontos) 5) Mostre a árvore de chamadas recursivas (pilha de recursão) e o resultado do algoritmo abaixo para a entrada inicial $X(4)$

```
1 int X( int n) {
2    if (n == 1 || n == 2) return n;
3    else return X( n-1) + n * X( n-2);
4 }
```

(30 pontos) 6) Uma sequência de N elementos V_i ORDENADOS, com N par, é dita balanceada quando a soma do maior elemento com o menor elemento é igual a soma do segundo maior elemento com o segundo menor elemento, que por sua vez é igual a soma do terceiro maior elemento com o terceiro menor elemento, e assim por diante. Escreva uma função RECURSIVA em C que decida se o vetor está balanceado ou não. Esta função deverá retornar 1 quanto estiver balanceado e 0 quando não estiver. Implemente quaisquer funções auxiliares que achar necessário.

A função tem o seguinte protótipo (e não pode ser modificado):

```
int balanceada(int *vetor, int inicio, int fim)
```

- $2 \leq N \leq 1000$
- $-1000 \leq V_i \leq 1000$

Exemplo de vetor balanceado:

```
1 3 7 11 15 17
```

Exemplo de vetor não balanceado:

```
3 3 5 8 9 10 11 27 32 41 45 51
```

(30 pontos) 7) Fazer um programa em C para ler uma sequência (não necessariamente ordenada) com uma quantidade arbitrária de números inteiros positivos do teclado e armazená-los em um vetor v. O último número lido é o zero, o qual não deve fazer parte dos valores de v. Além desses, mais um único valor deve ser lido, o que representa o limite de soma. Após a leitura dos dados, o programa deve imprimir apenas os últimos números de cada subsequência de v cuja soma de seus valores ultrapassa minimamente o limite de soma. Uma vez ultrapassado esse limite, uma nova subsequência deve ser iniciada a partir do valor que segue o último da subsequência identificada no momento. Ao final do processamento completo seu programa deverá imprimir o valor que ultrapassa minimamente o limite da soma ao contrário que da forma que é processado, e esses números NÃO podem ser armazenados em um vetor. Veja um exemplo de execução abaixo:

Exemplo de Entrada

```
33 51 23 94 66 28 11 73 19 8 31 0
90
```

Saída para o exemplo

```
19 28 94 23
```

No exemplo acima, o valor 23 é o último da subsequência 33 51 23, cuja soma é 107, a qual minimamente ultrapassa o limite de soma 90, sendo assim deverá ser impresso por último. Com isso, logo depois do valor 23, tem início a verificação da soma dos valores de uma nova subsequência que começa com o número 94. De cara, só o 94 já ultrapassa o limite 90, sendo o penúltimo número a ser impresso, e assim por diante até que todo o vetor v seja processado.

Note que você NÃO pode iniciar o processamento pelo fim do vetor v, pois, desta forma, o resultado seria diferente.