Programação Imperativa 2021/2022 (CC1003), DCC/FCUP

Exame da Época Normal

1 Considere um conjunto de 10 pontos no plano $\mathbb{R}^2$, (x_i, y_i), com $i = 0, \ldots, 9$. Escreva um programa que lê uma listas de 10 pontos da entrada padrão (dados pelos valores x_i e y_i), e que imprime na saída padrão o centroide de tais pontos, ou seja o ponto cujas coordenadas x_c e y_c são dadas por

$$x_c = \frac{\sum_{i=0}^{9} x_i}{10},\ y_c = \frac{\sum_{i=0}^{9} y_i}{10}.$$

Para obter cotação completa deste exercício, deve usar uma estrutura para representar o tipo de dados associado a um ponto.
O programa deve imprimir apenas os valores das coordenadas do centroide **com duas casas decimais**, separadas por uma vírgula e entre parênteses redondos (veja o exemplo de código no editor).

```
#include <stdio.h>

/* Completar com definições de estruturas úteis para o programa */

int main(void) {
     /* Completar este programa */

    printf("(%.2f,%.2f)", x_c, y_c);
    return 0;
}
```

2 Escreva um programa que lê um carater `c` da entrada padrão. O programa deve também ler o conteúdo das linhas do ficheiro de texto "texto.txt". O programa deve escrever na saída padrão o número de vezes que o carater `c` aparece em cada linha do ficheiro "texto.txt". Caso o carater `c` seja uma letra, a contagem das ocorrências do carater `c` em cada linha do ficheiro deve ser feita sem distinguir entre letras maiúsculas e letras minúsculas.
O conteúdo do ficheiro de texto é o seguinte:

```
LA DIVINA COMMEDIA
di Dante Alighieri
INFERNO
CANTO 1

Nel meZZo del caMmin di nostra vita
mi ritrovai per uNA SELVA oscura,
che la diritta via ERA SMarrita.
```

Portanto, se o carater lido da entrada padrão for:

```
a
```

o output do programa deve ser:

```
3
2
0
1
0
3
4
6
```

3 Escreva um programa que lê linhas da entrada padrão até encontrar end-of-file. O programa deve guardar o texto lido da entrada padrão num vetor de strings, onde cada linha corresponde a uma string do vetor. A seguir, o programa imprime na saída padrão todas as strings cujo tamanho seja maior ou igual ao tamanho médio das strings no vetor.
Por exemplo, se o input da entrada padrão for:

```
Lorem ipsum
dolor
sit amet
consectetur
```

o programa deve imprimir:

```
Lorem ipsum
consectetur
```

Para obter cotação máxima deste exercício, deve usar um vetor de apontadores para implementar o vetor de strings.

4a Complete o seguinte programa que lê uma sequência de inteiros positivos da entrada padrão terminada por 0 e que os guarda numa lista ligada, introduzindo cada elemento no início da lista. O programa deverá depois também imprimir na saída padrão a sequência de inteiros contidos na lista, a partir do início da lista até ao último elemento.
Por exemplo, se o input for:

```
1 3 8 19 7 3 5 0
```

o programa deve imprimir a seguinte linhas:

```
5 3 7 19 8 3 1
```

Para obter cotação completa deste exercício, deve guardar e manipular os dados usando uma lista ligada.

```
#include <stdio.h>
#include <stdlib.h>

struct node {
  int info;
```

```
  struct node *next;
};

struct node *novo_elemento(int info){
  /* completar esta função */
}

struct node *ins_inicio(struct node *first, int info){
  /* completar esta função */
}

void imprimir(struct node *first){
  /* completar esta função */
}

int main() {

  int i;
  struct node * first = NULL;
  while (1) {
    scanf("%d", &i);
    if(i == 0)
      break;

    first = ins_inicio(first, i);
  }
  imprimir(first);

  return 0;
}
```

4b Complete o seguinte programa que lê uma sequência de inteiros positivos da entrada padrão, terminada por 0, e que os guarda numa lista ligada, introduzindo cada elemento no início da lista.
O programa deverá depois remover da lista a primeira ocorrência do elemento com valor máximo. Por fim, o programa deve imprimir na saída padrão a sequência de inteiros contidos na lista (após remoção do elemento com valor máximo), a partir do início da lista até ao último elemento.
Por exemplo, se o input for:

```
1 3 8 19 7 3 5 0
```

o programa deve imprimir a seguinte linha:

```
5 3 7 8 3 1
```

Para obter cotação completa deste exercício, deve guardar e manipular os dados usando uma lista ligada.

```
#include <stdio.h>
#include <stdlib.h>
```

```
struct node {
  int info;
  struct node *next;
};

struct node *novo_elemento(int info){
  /* completar esta função */
}

struct node *ins_inicio(struct node *first, int info){
  /* completar esta função */
}

struct node *remover_maior(struct node *first){
  /* completar esta função */
}

void imprimir(struct node *first){
  /* completar esta função */
}

int main() {

  int i;
  struct node * first = NULL;
  while (1) {
    scanf("%d", &i);
    if(i == 0)
      break;

    first = ins_inicio(first, i);
  }

  first = remover_maior(first);
  imprimir(first);

  return 0;
}
```

Questionário de escolha múltipla

Considere a seguinte definição incompleta de uma função que implementa **ordenação por seleção**:

```
void select_sort(int vec[], int n) {
  int i, j;
  for(i = 0; i < n; i++) {
    int imin = i; // índice inicial do mínimo
```

```
    for(j = i+1; j < n; j++) {
      if(..?..) imin = j; // completar
    }
    // trocar o mínimo com vec[i]
    if(imin != i) {
      int temp = vec[i];
      vec[i] = vec[imin];
      vec[imin] = temp;
    }
  }
}
```

Qual das alternativas completa a condição em falta?
(a) `vec[j] < vec[i]`
(b) `vec[j] > vec[imin]`
(c) `vec[j] < vec[imin]`

Considere a seguinte função (incompleta) para a pesquisa binária de um valor `x` numa variável indexada `vec[]` ordenada com tamanho `n`.

```
int pesquisa_bin(int vec[], int n, int x) {
  int i = 0, j = n-1;
  while (i <= j)  {
    int k = i + (j-i)/2; // índice do meio
    if(vec[k] == x)
      return k;   // encontrou
    else if (x > vec[k])
      ??????;  // instrução A
    else
      ??????;  // instrução B
  }
  return -1;   // não encontrou
}
```

Como deve completar o programa para implementar corretamente a pesquisa binária?
(a) instrução A: `i=k+1;`
instrução B: `j=k-1;`
(b) instrução A: `k=i+1;`
instrução B: `k=j-1;`
(c) instrução A: `i=k-1;`
instrução B: `j=k+1;`

Considere as seguintes declarações de variáveis:

```
int i, *p, *q;
```

Indique qual das seguintes atribuições **não é** legal.

(a) `*p = i;`
(b) `*p = &i;`
(c) `i = *q;`

Considere a seguinte função (incompleta) que copia a cadeia de carateres em `origem` para `dest`

```
void copiar(char *dest, char *origem) {
   while(*origem != '\0')
      ??????;     // instrução em falta
   *dest = '\0';  // colocar terminador
}
```

Como deve completar a instrução em falta?
(a) `*dest++ = *origem++;`
(b) `*origem++ = *dest++;`
(c) `*dest = origem++;`

Considere as seguinte definições de estruturas, que são usadas para definir a informação de um contacto numa agenda. Cada registo contém informações sobre o nome, o número de telefone e a data de nascimento do contacto correspondente:

```
typedef struct data{
  int dia, mes, ano;
} DATA;

typedef struct contact{
  char nome[30];
  int telefone;
  DATA nascimento;
} CONTACTO;
```

Qual das seguintes expressões acede ao valor do mês da data de nascimento do contacto cujo apontador é definido como `CONTACTO *cont1;`?
(a) `cont1->nascimento.mes`
(b) `cont1->nascimento->mes`
(c) `cont1.nascimento.mes`

Considere um programa escrito em C usando vários ficheiros header e source.
Qual das seguintes instruções declara uma variável inteira `i` sem a definir?
(a) `int i;`
(b) `extern int i;`
(c) `define i`

Considere um programa escrito em C usando vários ficheiros header e source.
Quando queremos invocar uma dada função em vários ficheiros, é conveniente:
(a) definir a função num ficheiro header
(b) declarar o protótipo da função num ficheiro source e a definição num ficheiro header, que inclui o ficheiro

source
(c) declarar o protótipo da função num ficheiro header e a definição num ficheiro source, que inclui o ficheiro header

Considere as seguintes instruções

```
int a[3];
*a=1;
*(a+1)=12;
*(a+2)=0;
free(a);
```

Neste caso, a instrução `free(a)`
(a) liberta a memória ocupada pelo vetor `a`
(b) inicializa com valor `0` todos os elementos do vetor `a`
(c) tem um comportamento indefinido

Qual das seguintes instruções pode ser usada para reservar a memória necessária para um vetor que contem 4 valores inteiros:
(a) `int vec = calloc(4,sizeof(int));`
(b) `int *vec = malloc(4*sizeof(int));`
(c) `int *vec = malloc(4);`

Seja `fp` um apontador para um objeto de tipo `FILE`. Qual das seguintes instruções é equivalente à instrução `rewind(fp)`?
(a) `fseek(fp, OL, SEEK_SET);`
(b) `fseek(fp, OL, SEEK_CUR);`
(c) `fstart(fp, 0L, SEEK_SET);`

Considere a seguinte definição de estrutura

```
struct data{
  int dia, mes, ano;
};
```

e a seguinte declaração de um vetor de estruturas

```
struct data vec[10];
```

Qual das seguintes instruções pode ser usada para gravar o conteúdo do vetor `vec` no ficheiro binário apontado pelo apontador `fp`?
(a) `fwrite(vec, sizeof(struct data),10, fp);`
(b) `fread(vec, sizeof(vec),10, fp);`
(c) `fwrite(vec, sizeof(vec),10, fp);`