MAC 2166 Introdução à Computação para Engenharia

PROVA 3

QUESTÃO 1.

Simule a execução do programa abaixo, destacando a sua saída. A saída do programa consiste de tudo que resulta dos comandos `printf`.

```
# include <stdio.h>
void f1(int v[4], int k){
  int i;
  for (i = 0; i < 4; i++)
    v[i] = k + i;
}

void f2(int m[2][2], int v[4]){
  int i, j;
  for (i = 0; i < 2; i++)
    for (j = 0; j < 2; j++)
      m[i][j] = v[2*i+j];
}

int f3(int k){
  int i, j;
  i = 2; j = 1;
  k = k + 2;
  return k;
}

int f4(int *k){
  int i, j;
  i = 2; j = 1;
  *k = *k + 2;
  return *k;
}

int main(){
  int nusp, k, n, i, j;
  int a[2][2], v[4];

  printf ("Entre com seu no. USP: ");
  /* na linha abaixo use o numero USP */
  scanf ("%d", &nusp);
  printf ("nusp = %d\n", nusp);

  k = nusp / 10;
  n = k % 5;
  printf ("1: k=%d n=%d\n", k, n);

  for (i = 0; i < 4; i++)
    v[i] = i;

  printf("2: %d %d %d %d\n",v[0],v[1],v[2],v[3]);

  f1(v,n);

  printf("3: %d %d %d %d\n",v[0],v[1],v[2],v[3]);

  f2(a,v);

  printf("4: %d %d\n",a[0][0], a[0][1]);
  printf("5: %d %d\n",a[1][0], a[1][1]);

  i = 0; j = 1;
  i = f3(j);

  printf("6: i=%d j=%d\n",i, j);

  i = 0; j = 1;
  v[i] = f3(v[j]);

  printf("7: %d %d %d %d\n",v[0],v[1],v[2],v[3]);

  i = 2; j = 3;
  i = f4(&j);

  printf("8: i=%d j=%d\n", i, j);
```

```
    i = 2; j = 3;
    v[i] = f4(&v[j]);

    printf ("9: %d %d %d %d\n",v[0],v[1],v[2],v[3]);

    return 0;
}
```

SOLUÇÃO.

A resposta depende, essencialmente, do penúltimo dígito do número USP. Teste com o seu no. USP e compare a resposta.

```
(0) penúltimo dígito % 5 == 0

Entre com seu no. USP: 101
nusp = 101
1: k=10 n=0
2: 0 1 2 3
3: 0 1 2 3
4: 0 1
5: 2 3
6: i=3 j=1
7: 3 1 2 3
8: i=5 j=5
9: 3 1 5 5

(1) penúltimo dígito % 5 == 1

Entre com seu no. USP: 111
nusp = 111
1: k=11 n=1
2: 0 1 2 3
3: 1 2 3 4
4: 1 2
5: 3 4
6: i=3 j=1
7: 4 2 3 4
8: i=5 j=5
9: 4 2 6 6

(2) penúltimo dígito % 5 == 2

Entre com seu no. USP: 121
nusp = 121
1: k=12 n=2
2: 0 1 2 3
3: 2 3 4 5
4: 2 3
5: 4 5
6: i=3 j=1
7: 5 3 4 5
8: i=5 j=5
9: 5 3 7 7


(3) penúltimo dígito % 5 == 3

Entre com seu no. USP: 131
nusp = 131
1: k=13 n=3
2: 0 1 2 3
3: 3 4 5 6
4: 3 4
5: 5 6
6: i=3 j=1
7: 6 4 5 6
8: i=5 j=5
9: 6 4 8 8

(4) penúltimo dígito % 5 == 4

Entre com seu no. USP: 141
nusp = 141
1: k=14 n=4
2: 0 1 2 3
3: 4 5 6 7
4: 4 5
5: 6 7
6: i=3 j=1
7: 7 5 6 7
8: i=5 j=5
9: 7 5 9 9
```

QUESTÃO 2.

Faça um programa que lê:

1. um número inteiro `n`, `0<n<1000`;
2. uma seqüência de `n` números inteiros, sem repetição;
3. um número inteiro `m`, `0<m<1000`;
4. uma seqüência de `m` números inteiros, sem repetição

e imprime os números que aparecem nas duas sequências. Exemplo:

```
Entrada:

5
4  -7   24   5   8
3
5   6   -7

Saída:

O número 5 aparece nas duas sequencias.
O número -7 aparece nas duas sequencias.
```

SOLUÇÃO.

```
/*
 * VERSÃO 1:  usa apenas um vetor e a saída exibida na tela mistura
 *   mensagens de entre-com-número e mensagens de achei-o-número.
 *
 */
#include <stdio.h>
#define MAX 1000

int main() {
  int n;      /* numero de elementos na 1a. sequencia */
  int v[MAX]; /* numeros na primeira sequencia */
  int m;      /* numero de elementos  na 2a. sequencia */
  int num;    /* usado para ler a 2a. sequencia */
  int i;      /* usada como indicedo vetor v */
  int j;      /* usada como contador de elementos na 2a. sequencia */
  int achou;  /* variavel que indica se um numero foi encontra em v */

  /* 1. leia o numero de elementos da 1a. sequencia */
  printf("Entre com n: ");
  scanf("%d", &n);

  /* 2. leia a 1a. sequencia */
  for (i = 0; i < n; i++)
    {
      printf("Entre com o %io. numero da 1a. sequencia: ", i+1);
      scanf("%d", &v[i]);
    }


  /* 3. leia o numero de elementos da 2a. sequencia */
  printf("Entre com m: ");
  scanf("%d", &m);

  /* 4. le e trata um a um os numeros na 2a. sequencia */
  for (j = 0; j < m; j++)
    {
      printf("Entre com o %io. numero da 2a. sequencia: ", j+1);
      scanf("%d", &num);

      /* procure num na 1a. sequencia */
      achou = 0;
      for (i = 0; i < n && achou == 0; i++)
        {
           if (v[i] == num)  achou = 1;
        }

      if (achou == 1)
        {
           printf("O numero %d aparece nas duas sequencias.\n", num);
        }
    }

  return 0;
}


NA TELA:
Entre com n: 5
Entre com o 1o. numero da 1a. sequencia: 4
```

```
Entre com o 2o. numero da 1a. sequencia: -7
Entre com o 3o. numero da 1a. sequencia: 24
Entre com o 4o. numero da 1a. sequencia: 5
Entre com o 5o. numero da 1a. sequencia: 8
Entre com m: 3
Entre com o 1o. numero da 2a. sequencia: 5
O numero 5 aparece nas duas sequencias.
Entre com o 2o. numero da 2a. sequencia: 6
Entre com o 3o. numero da 2a. sequencia: -7
O numero -7 aparece nas duas sequencias.
```

```
/*
 * VERSÃO 2:  usa dois vetores e não mistura
 *   na tela as mensagens de entre-com-número e mensagens de achei-o-número.
 *
 */
#include <stdio.h>
#define MAX 1000

int main() {
  int n;      /* numero de elementos na 1a. sequencia */
  int v[MAX]; /* numeros na 1a. sequencia */
  int m;      /* numero de elementos  na 2a. sequencia */
  int w[MAX]; /* numeros na 2a. sequencia */
  int i;      /* usada como indicedo vetor v */
  int j;      /* usada como contador de elementos na 2a. sequencia */
  int achou;  /* variavel que indica se um numero foi encontra em v */

  /* 1. leia o numero de elementos da 1a. sequencia */
  printf("Entre com n: ");
  scanf("%d", &n);

  /* 2. leia a 1a. sequencia */
  for (i = 0; i < n; i++)
    {
      printf("Entre com o %io. numero da 1a. sequencia: ", i+1);
      scanf("%d", &v[i]);
    }

  /* 3. leia o numero de elementos da 2a. sequencia */
  printf("Entre com m: ");
  scanf("%d", &m);

  /* 4. leia a 2a. sequencia */
  for (j = 0; j < m; j++)
    {
      printf("Entre com o %io. numero da 1a. sequencia: ", j+1);
      scanf("%d", &w[j]);
    }

  /* 5. verifique quais numeros estao em ambas as sequencias */
  for (i = 0; i < n; i++)
    {
      /* procure num na 1a. sequencia */
      achou = 0;
      for (j = 0; j < m && achou == 0; j++)
        {
           if (w[j] == v[i])  achou = 1;
        }

      if (achou == 1)
        {
           printf("O numero %d aparece nas duas sequencias.\n", v[i]);
        }
    }

  return 0;
}
```

```
NA TELA:
Entre com n: 5
Entre com o 1o. numero da 1a. sequencia: 4
Entre com o 2o. numero da 1a. sequencia: -7
Entre com o 3o. numero da 1a. sequencia: 24
Entre com o 4o. numero da 1a. sequencia: 5
Entre com o 5o. numero da 1a. sequencia: 8
Entre com m: 3
Entre com o 1o. numero da 1a. sequencia: 5
Entre com o 2o. numero da 1a. sequencia: 6
Entre com o 3o. numero da 1a. sequencia: -7
O numero -7 aparece nas duas sequencias.
O numero 5 aparece nas duas sequencias.
```

QUESTÃO 3

Esta questão consiste na implementação de duas funções. Uma das funções é o `main` e a outra é uma função de nome `afunde_destroyer`. Ambas as funções são simplificações **muito grandes** de alguns trechos de código que você escreveu no seu EP3.

Considere uma região retangular do mar Sei-lá-o-nome, que você já conhece. O mar é descrito através de um mapa representado no computador por uma *matriz de caracteres* que contém informações sobre o conteúdo de cada posição da região. Essa região tem no máximo 18 linhas e no máximo 18 colunas. Abaixo está um `mapa' que ilustra uma região.

```
   MAPA DA REGIAO
   no. linhas = 5  no. colunas = 8
     1   2   3   4   5   6   7   8
   +---+---+---+---+---+---+---+---+
 1 | D |   |   |   | D |   | D | D |
   +---+---+---+---+---+---+---+---+
 2 |   | D |   |   | D |   |   |   |
   +---+---+---+---+---+---+---+---+
 3 |   |   |   |   |   |   | D | D |
   +---+---+---+---+---+---+---+---+
 4 | D | D |   |   | D |   |   |   |
   +---+---+---+---+---+---+---+---+
 5 |   |   |   | D |   |   |   |   |
   +---+---+---+---+---+---+---+---+
```

O **único** tipo permitido de embarcação nessa região é o *destroyer*:

```
+---+---+
| D | D |
+---+---+
```

Os destroyers podem estar dispostos no mapa na horizontal, na vertical ou na diagonal. Entretanto, eles **não** podem se tocar. Os protótipos de algumas funções desta questão usam as seguintes declarações:

```
#define MAXLINHAS     20
#define MAXCOLUNAS    20
```

item (a) Escreva uma função de protótipo

```
void afunde_destroyer(int lin_d, int col_d,
                      int no_linhas, int no_colunas,
                      char mapa[MAXLINHAS][MAXCOLUNAS]);
```

que recebe uma posição e uma matriz `mapa` cujas linhas de 1 a `no_linhas` e colunas de 1 a `no_colunas` representam uma região. A posição `(lin_d,col_d)` contém uma das partes de um destroyer. A função `afunda' o destroyer, ou seja, coloca o caractere `'*'` na posição de `(lin_d,col_d)` e troca o outro caractere `'D'` do detroyer por `'d'`.

SOLUÇÃO.

```
/*
 * VERSÃO 1:  usa uma moldura, não
 *    utiliza os parâmetros no_linhas e no_colunas.
 */
void afunde_destroyer (int lin_d, int col_d,
                  int no_linhas, int no_colunas,
                  char mapa[MAXLINHAS][MAXCOLUNAS])
{
  int i, j;

  mapa[lin_d][col_d] = '*';

  for (i = -1; i <= 1; i++)
    for (j = -1; j <= 1; j++)
      if (mapa[lin_d+i][col_d+j] == 'D')
        mapa[lin_d+i][col_d+j] = 'd';

}


/*
 * VERSÃO 2:  não usa uma moldura,
 *    utiliza os parâmetros no_linhas e no_colunas.
 */
void afunde_destroyer (int lin_d, int col_d,
                  int no_linhas, int no_colunas,
                  char mapa[MAXLINHAS][MAXCOLUNAS])
{
  int i, j;
  int linha, coluna;
```

```
      mapa[lin_d][col_d] = '*';

      for (i = -1; i <= 1; i++)
        for (j = -1; j <= 1; j++)
          {
            linha = lin_d+i; coluna = col_d+j;
            if (1 <= linha && linha <= no_linhas && 1 <= coluna && coluna <= no_colunas)
               if (mapa[linha][coluna] == 'D')
                  mapa[linha][coluna] = 'd';
    }
```

Para escrever a função `main` pedida no item~(b) você **deve** usar as seguinte funções **sem** escrevê-las. Suponha que, como ocorreu no EP3, lhe é dada uma função de protótipo

```
        void leia_mapa (int *no_linhas, int *no_colunas,
                        char mapa[MAXLINHAS][MAXCOLUNAS]);
```

que lê uma uma matriz `mapa` de `*no_linhas` linhas e `*no_colunas` colunas. Após a leitura a região retangular está representada nas linhas de 1 a `*no_linhas` e colunas de 1 a `*no_colunas` da matriz `mapa`. **Não** se preocupe como e de onde a matriz é lida.

Suponha ainda que lhe é dada uma função de protótipo

```
        void escreva_mapa_tela(int no_linhas, int no_colunas,
                               char mapa[MAXLINHAS][MAXCOLUNAS]);
```

que recebe uma matriz `mapa` de `no_linhas` linhas e `no_colunas` colunas e que escreve na tela a região representada pela matriz `mapa` exibindo **todos** os caracteres da matriz.

item (c) Escreva uma função main que lê o mapa de um região, um número inteiro positivo `no_tiros` e uma seqüência de `no_tiros` posições do mapa que devem ser destruídas por tiros. Para cada uma dessas `no_tiros` posições o seu programa deve indicar se foi atingido um destroyer ou água. Se um destroyer é atingido ele deve ser afundado. Se uma posição que contém água é atingida, o caractere 'a' deve ser colocado nessa posição. O mapa resultante deve ser impresso ao final do programa. A sua função main deve usar **obrigatoriamente** as funções

`leia_mapa`, `escreva_mapa_tela` e `afunde_destroyer`.

Você pode usar a função `afunde_destroyer` do item (a) **mesmo que não as tenha feito**. Sinta-se à vontade para escrever qualquer outra função que desejar. Por exemplo, o seu programa pode ter a seguinte saída na tela, onde os números após dois pontos (':') indicam os valores lidos pelo programa. Note que as posições do tiros são **lidas** e não geradas por uma função como no EP3.

```
Entre com o numero de tiros:  4
Entre com a posicao do tiro (linha,coluna): 2  2
Tiro na posicao (2,2) acertou um DESTROYER.
Entre com a posicao do tiro (linha,coluna): 5  6
Tiro na posicao (5,6) acertou AGUA.
Entre com a posicao do tiro (linha,coluna): 1  1
Tiro na posicao (1,1) acertou AGUA.
Entre com a posicao do tiro (linha,coluna): 4  1
Tiro na posicao (4,1) acertou um DESTROYER.
    MAPA DA REGIAO
    no. linhas = 5  no. colunas = 8
      1   2   3   4   5   6   7   8
    +---+---+---+---+---+---+---+---+
  1 | a |   |   |   | D |   | D | D |
    +---+---+---+---+---+---+---+---+
  2 |   | * |   |   | D |   |   |   |
    +---+---+---+---+---+---+---+---+
  3 |   |   |   |   |   |   | D | D |
    +---+---+---+---+---+---+---+---+
  4 | * | d |   |   | D |   |   |   |
    +---+---+---+---+---+---+---+---+
  5 |   |   |   | D |   | a |   |   |
    +---+---+---+---+---+---+---+---+
```

SOLUÇÃO.

```
/*
 * A solucao abaixo usa uma moldura.
 *
 */
void emoldure (int no_linhas, int no_colunas, char mapa[MAXLINHAS][MAXCOLUNAS])
{
  int i; /* contador de linhas */
  int j; /* contador de colunas */

  for (i = 0; i <= no_linhas+1; i++)
    for (j = 0; j <= no_colunas+1; j++)
      if (i == 0 || i == no_linhas+1 || j == 0 || j == no_colunas+1)
        mapa[i][j] = MOLDURA;
}
```

```
int main()
{
  char mapa[MAXLINHAS][MAXCOLUNAS];
  int no_linhas;  /* numero de linhas do mapa */
  int no_colunas; /* numero de colunas do mapa */
  int no_tiros;   /* numero de tiros */
  int i;          /* contador de numero de tiros */
  int l_tiro;     /* linha do tiro */
  int c_tiro;     /* coluna do tiro */

  /* 1. leia o mapa */
  leia_mapa(&no_linhas, &no_colunas, mapa);

  /* 2. emoldure o mapa */
  emoldure(no_linhas, no_colunas, mapa);

  /* 3. escreva o mapa na tela*/
  escreva_mapa_tela(no_linhas, no_colunas, mapa);

  /* 4. leia numero de tiros */
  printf("Entre com o numero de tiros: ");
  scanf("%d", &no_tiros);

  /* 5. trata os tiros um a um */
  for (i = 0; i < no_tiros; i++)
    {
      printf("Entre com a posicao do tiro (linha,coluna): ");
      scanf("%d %d", &l_tiro, &c_tiro);

      printf("Tiro na posicao (%d,%d) acertou", l_tiro, c_tiro);
      if (mapa[l_tiro][c_tiro] == 'D')
        {
          printf(" um DESTROYER.\n");
          afunde_destroyer(l_tiro, c_tiro, no_linhas, no_colunas, mapa);
        }
      else
        {
          printf(" AGUA.\n");
          mapa[l_tiro][c_tiro] = 'a';
        }
    }

  /* 6. escreva o mapa na tela*/
  escreva_mapa_tela(no_linhas, no_colunas, mapa);

  return 0;
}
```
