Terceira prova de MAC2166
Grande área Elétrica – 1 de julho de 2025

Computação (turma 01) (__) Elétrica1 (turma 02) (__) Elétrica2 (turma 03) (__)

Nome do aluno: _____

Assinatura: _____

Nº USP: _____ Professor: _____

Instruções:

1. Não utilize recursos da linguagem C que ainda não foram vistos em aula.
2. Não destaque as folhas deste caderno.
3. A prova pode ser feita a lápis, mas **PRECISA** ter as **resoluções claras, organizadas, endentadas e legíveis.** Cuidado com a legibilidade.
4. Responda cada questão dentro do espaço indicado. Pode continuar no verso da página, mas deixe indicado.
5. A prova consta de 4 questões. Verifique antes de começar a prova se o seu caderno de questões está completo.
6. Não é permitido o uso de folhas avulsas para rascunho.
7. Não é necessário apagar rascunhos no caderno de questões.
8. Não é permitido o uso de equipamentos eletrônicos, inclusive celular.
9. Não é permitido a consulta a livros, apontamentos ou colegas.

Não escreva nesta parte da folha

Questão	Valor	Nota
1	$2,4$	
2	$2,4$	
3	$2,4$	
4	$2,8$	
Total	dez	

BOA PROVA!!!

Questão 1 (valor: 2,4 pontos)
Simule a execução do programa abaixo, destacando a sua **saída**. A saída do programa consiste de tudo que resulta das chamadas a `printf`.

```
1  #include <stdio.h>
2
3  int Bah( int x, int n, int y[], int *z ) {
4    int b,e,m, k=*z;
5    b = 0; e = n;
6    while ( 2 <= e - b ) {
7      m = (b + e) / 2; k++;
8      printf( "%d:  %d  %d  %d  %d\n", k, b,e,m, y[m] );
9      if ( x < y[m] ) e = m;
10     else b = m;
11   }
12   m = (b + e) / 2; k++;
13   printf( "%d:  %d  %d  %d  %d\n", k, b,e,m, y[m] );
14   *z = k;
15   return (x==y[b]);
16 }
17
18 void Ahm( int S[], int n ) {
19   int i;
20   for ( i = 0; i < n; i++)
21     printf( "%d ", S[i] );
22   printf( "\n" );
23   return;
24 }
25
26
27 int main() {
28    int nusp, d, a;
29    int x, y, z[100];
30
31    printf("Digite o seu numero USP:\n");
32    scanf( "%d", &nusp );
33    d = nusp % 10;
34    if (d > 7) d = d - 2;
35    printf("d = %d\n", d);
36    for(y = 0; y < 8; y++)
37      z[y] = d + y;
38    Ahm( z, 8 );
39    x = 0;
40    a = Bah( d + 5, 8, z, &x );
41    x++;  printf( "%d: %d\n", x, a );
42    a = Bah( d + 8, 8, z, &x );
43    x++;  printf( "%d: %d\n", x, a );
44    return 0;
45 }
```

Para efeito de correção *só será considerada a <u>saída</u>* do programa. Você pode usar a tabela abaixo como bem entender.

main					Bah						
nusp	d	a	x	y	x	y	b	e	m	k	z

saída

Questão 2 (valor: $2,4$ pontos) Podemos representar um tabuleiro do jogo campo minado (minesweeper) através de uma matriz binária em que o valor 0 indica uma posição livre e 1 indica a presença de uma mina. Ao se inspecionar uma casa deste tabuleiro, deseja-se primeiro saber se há nela uma mina. Não havendo, deseja-se ainda saber quantas minas existem ao redor desta casa (na horizontal, vertical ou nalguma diagonal). Escreva uma **função** em C que recebe como parâmetros um inteiro `n`, uma matriz binária `M` representando um tabuleiro `n` por `n`, a linha `lin` e a coluna `col` de uma casa a ser inspecionada. Caso haja uma mina na casa de linha `lin` e coluna `col`, a função deve devolver -1. Caso contrário, deve-se devolver o número de minas ao redor desta casa. As linhas e colunas da matriz são enumeradas a partir de 0. Por exemplo, para a matriz `M` seguinte e `lin` $= 0$, a função deve devolver -1, 2 e 0 para `col` $= 0, 1, 2$, respectivamente.

```
1 0 0
1 0 0
0 0 0
```

Questão 3 (valor: 2,4 pontos) Escreva em C um **programa** que lê da entrada uma string, sabidamente formada apenas por dígitos, e imprime na saída a string obtida pela ordenação dos dígitos em ordem decrescente. Por exemplo, para a string de entrada `58130`, o programa deve produzir a string `85310` e imprimi-la.

Questão 4 (valor: 2,8 pontos) Considere um jogo de Tetris implementado em C em que o tabuleiro é representado por uma matriz de inteiros de 20 linhas por 11 colunas. Nessa matriz, 0 representa espaço vazio, 1 representa bloco, 2 representa borda lateral (parede) e 3 representa borda do fundo (chão). Esses valores são impressos na tela seguindo a associação: 0 → ' ', 1 → '#', 2 → '|' e 3 → '-'. As constantes abaixo estão presentes na implementação:

```
#define LINHAS     20
#define COLUNAS    11
#define VAZIO     ' '
#define HASHTAG   '#'
#define PAREDE    '|'
#define CHAO      '-'
#define NUM_VAZIO   0
#define NUM_HASHTAG 1
#define NUM_PAREDE  2
#define NUM_CHAO    3
```

(a) Escreva uma **função** em C de nome `apagaTopoTabuleiro` que recebe como parâmetro a matriz de inteiros representando o tabuleiro do jogo, contabiliza todos os blocos (`NUM_HASHTAG`) no topo das colunas, substitui esses blocos por espaço vazio (`NUM_VAZIO`) e retorna o valor inteiro $\lfloor \frac{blocos}{2} \rfloor$, onde $\lfloor x \rfloor$ retorna a parte inteira de x (Ex.: $\lfloor \frac{5}{2} \rfloor = 2$, $\lfloor \frac{2}{2} \rfloor = 1$). Por exemplo, caso a função receba como parâmetro uma matriz representando o tabuleiro à esquerda abaixo, ela modificará a matriz de modo que ela passe a representar o tabuleiro à direita e retornará o valor 3 (os blocos no topo das colunas 1, 3, 4, 5, 6 e 7 foram removidos. Como foram 6 blocos e $\lfloor \frac{6}{2} \rfloor = 3$, a função retorna 3). Na sua resposta, use preferencialmente as constantes listadas acima.

```
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|         |                |         |
|   #     |                |         |
|   ###   |                |   #     |
|   ###   |                |   ###   |
|#  ####  |                |   ###   |
|  #####  |                |   ####  |
-----------                -----------
```

(b) Escreva uma **função** em C de nome `compactaTabuleiro` que recebe como parâmetro a matriz de inteiros representando o tabuleiro do jogo, e move todos os blocos (`NUM_HASHTAG`) de todas as colunas o máximo possível para baixo. A função não deve retornar nada. Por exemplo, caso a função receba como parâmetro uma matriz representando o tabuleiro à esquerda abaixo, ela modificará a matriz de modo que ela passe a representar o tabuleiro à direita (os blocos nas colunas 1, 4, 8 e 9 foram movidos para baixo). Na sua resposta, use preferencialmente as constantes listadas no início do enunciado da questão.