# MAC 2166 – Introdução à Computação

Engenharia Elétrica e de Computação
Primeiro Semestre de 2022

Prova de Recuperação – 26 de julho de 2022

Nome do aluno: _____

NUSP: _____ Turma: _____

Assinatura: _____

**Instruções:**

1. Não destaque as folhas deste caderno.
2. Preencha o cabeçalho acima.
3. A prova pode ser feita a lápis. Cuidado com a legibilidade.
4. A prova consta de **5 questões**. Verifique antes de começar a prova se o seu caderno de questões está completo.
5. Não é permitido o uso de folhas avulsas para rascunho.
6. Não é permitido o uso de artefatos eletrônicos.
7. Não é permitido a consulta a livros, apontamentos ou colegas.
8. Não é necessário apagar rascunhos no caderno de questões.

## DURAÇÃO DA PROVA: 2 horas

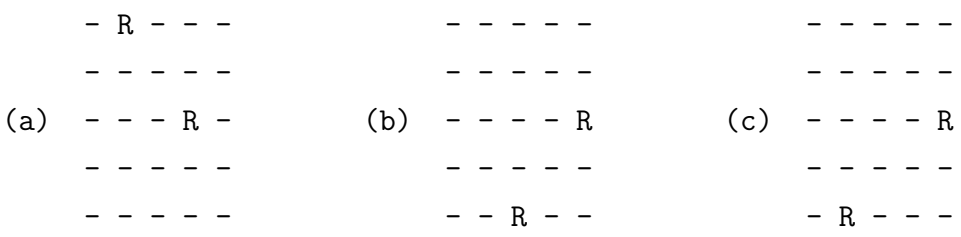
| Questão | Nota |
|---|---|
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| Total | |

**Q1 (3.0 pontos)** Simule a execução do programa abaixo, usando seu número USP como entrada. Basta indicar a saída do programa, no espaço indicado.

```
#include <stdio.h>
#define BASE 10

void f(int NUSP, int freq[BASE], int *a, int *b);
void p(int freq[BASE]);

int main()
{
  int NUSP, freq[BASE], a, b;

  scanf("%d", &NUSP);
  printf("0: %d\n", NUSP);
  f(NUSP, freq, &a, &b);
  printf("a = %d / b = %d\n", a, b);
  p(freq);
  NUSP = NUSP / 10000 + (NUSP % 10000) * 10000;
  printf("1: %d\n", NUSP);
  f(NUSP, freq, &a, &b);
  printf("a = %d / b = %d\n", a, b);
  p(freq);

  return 0;
}

void f(int NUSP, int freq[BASE], int *a, int *b) {
  int i, d, D = 0;
  for (i = 0; i < BASE; ++i)
    freq[i] = 0;
  while (NUSP > 0) {
    d = NUSP % BASE;
    if (freq[d] == 0)
      D++;
    freq[d]++;
    NUSP = NUSP / 10;
  }
  *a = D; *b = BASE - D;
}

void p(int freq[BASE]) {
  int i;

  for (i = 0; i < BASE; ++i)
    if (freq[i] > 0)
      printf("%d: %d\n", i, freq[i]);
}
```

Saída:

**Q2 (2.0 pontos)** Nesta questão e nas próximas, consideramos configurações de rainhas em um tabuleiro de xadrez $N$ por $N$ (com $N \leq 20$). Considere as seguintes três configurações do tabuleiro $5 \times 5$:




Podemos indexar as linhas e colunas do tabuleiro pelos inteiros $0, 1, \ldots, 4$. Fazendo assim, em (a), as rainhas estão nas posições $(0, 1)$ e $(2, 3)$. Em (b) elas estão em $(2, 4)$ e $(4, 2)$ e em (c) elas estão em $(2, 4)$ e $(4, 1)$. Nas configurações (a) e (b), as rainhas se atacam. Na configuração (c) as rainhas não se atacam.

Escreva uma função de protótipo

```
int ataca(int i, int j, int ii, int jj);
```

que devolve 1 (ou `TRUE`) se rainhas nas posições $(i, j)$ e $(ii, jj)$ se atacam e devolve 0 (ou `FALSE`) caso contrário.

**Q3 (2.0 pontos)** Podemos representar configurações com $N$ rainhas em um tabuleiro $N \times N$ por um vetor de inteiros com $N$ entradas. Por exemplo, as configurações

```
     - - - - R          - - R - -          - - R - -
     - - R - -          - - - - R          - - - - R
(a)  R - - - -     (b)  - R - - -     (c)  R - - - -
     - - - R -          - - - R -          - - - R -
     - R - - -          R - - - -          - R - - -
```

de 5 rainhas no tabuleiro $5 \times 5$ podem ser representadas pelos vetores

```
u = {4, 2, 0, 3, 1}
v = {2, 4, 1, 3, 0}
w = {2, 4, 0, 3, 1}
```

De fato, note que, em (a), as rainhas estão nas posições $(0, 4)$, $(1, 2)$, $(2, 0)$, $(3, 3)$ e $(4, 1)$, isto é, nas posições $(i, u[i])$ para $i = 0, \ldots, 4$. Em (b), as rainhas estão nas posições $(i, v[i])$ para $i = 0, \ldots, 4$ e analogamente para (c).

Note que nas configurações (a) e (b) nenhuma rainha ataca outra, enquanto que na configuração (c) as rainhas nas posições $(0, 2)$ e $(2, 0)$ se atacam (as rainhas nas posições $(1, 4)$ e $(4, 1)$ também se atacam).

Escreva uma função de protótipo

```
int valida(int v[NMAX], int N, int *a, int *b);
```

que recebe um vetor $v$ de inteiros que representa uma configuração de $N$ rainhas em um tabuleiro $N \times N$ e que devolve 1 (ou `TRUE`) se em $v$ não há rainhas (distintas) que se atacam mutuamente e devolve 0 (ou `FALSE`) caso contrário. Por exemplo, com os dois primeiros vetores no exemplo acima (que representam as configurações (a) e (b)) como parâmetro, sua função deve devolver `TRUE` e com o terceiro vetor como parâmetro sua função deve devolver `FALSE`. Ademais, no caso em que a configuração representada pelo vetor $v$ tiver rainhas que se atacam mutuamente, então os parâmetros $*a$ e $*b$ devem ser usados para devolver índices $i$ e $ii$ tais que as rainhas nas posições $(i, v[i])$ e $(ii, v(ii))$ atacam-se mutuamente. Por exemplo, a chamada

```
valida(w, 5, &i, &ii);
```

onde $w$ é o terceiro vetor no exemplo acima, pode resultar em $i$ assumir o valor 0 e $ii$ assumir o valor 2 (esses valores podem também ser 1 e 4, respectivamente).

Você deve usar a função da Questão **Q2**, mesmo que você não a tenha escrito.

**Q4 (2.0 pontos)** Escreva um programa que lê um inteiro $N$ e um vetor de $N$ inteiros representando uma configuração de $N$ rainhas em um tabuleiro $N \times N$ (como discutido na Questão **Q3**) e que imprime se a configuração é válida ou não (isto é, se não há rainhas que se atacam mutuamente ou se há). No caso de a configuração lida não ser válida, seu programa deve imprimir a posição de duas rainhas que se atacam mutuamente. Por exemplo, com a entrada

```
5
4 2 0 3 1
```

seu programa deve imprimir

```
Configuracao valida
```

Com a entrada

```
5
2 4 0 3 1
```

seu programa deve imprimir

```
Configuracao invalida:
Rainha na posicao (0, 2) ataca rainha na posicao (2, 0)
```

ou

```
Configuracao invalida:
Rainha na posicao (1, 4) ataca rainha na posicao (4, 1)
```

Seu programa deve usar a função da questão anterior, mesmo que você não a tenha escrito. Não é necessário reescrever aquela função aqui. Basta escrever o `main()` de seu programa.

**Q5 (2.0 pontos)** Escreva uma função de protótipo

```
void maiusculas(char s[NMAX]);
```

que recebe uma string em $s$ e que transforma cada letra minúscula em $s$ em sua letra maiúscula correspondente. Por exemplo, o trecho de código

```
scanf("%[^\n]", s);
maiusculas(s);
printf("%s\n", s);
```

executado com a entrada

```
It was the best of times, it was the worst of times,
```

deve imprimir

```
IT WAS THE BEST OF TIMES, IT WAS THE WORST OF TIMES,
```

**Não use nenhuma função de biblioteca em sua implementação de** `maiuscula()`.

***Dica.*** O trecho de código

```
c1 = 'b';
c2 = c1 + 'A' - 'a';
printf("c2 = %c\n", c2);
```

imprime

```
c2 = B
```

(Acima, supomos que `c1` e `c2` são variáveis to tipo char.)