Primeira prova de MAC2166

03 de abril de 2025 — turmas em C

NOME: _____

NUSP: _____

COMPUTAÇÃO (turma 01) (__) ELÉTRICA (turma 02) (__) ELÉTRICA (turma 03) (__)

Responda cada questão dentro do espaço indicado. **Se necessário, utilize o verso da folha**. A prova **pode ser resolvida a lápis** (exceto grafite HB), mas **PRECISA** ter as **resoluções claras, organizadas, endentadas e legíveis.** Não utilize recursos da linguagem não abordados em sala ou nas notas de aula.

O caderno de questões deve conter **QUATRO** questões. Verifique se seu caderno está completo.

1 Questão 1 (2 pontos):

Simule o programa abaixo e destaque na culuna *saída* da tabela seguinte o que o programa colocará na tela do computador. Caso queira, pode-se usar as colunas *nusp*, *a* e *b* e *c* como área de rascunho.

```
#include <stdio.h>
int main()
{
  int nusp, a, b, c;
  printf ("Entre com o seu no. USP: ");
  /* use o seu numero USP como dado de entrada */
  scanf ("%d", &nusp);
  printf ("nusp = %d\n", nusp);
  a = nusp % 10;
  b = 9 + a;
  c = 31 - a;
  printf("a = %d b = %d c = %d\n", a, b, c);
  while (c > 0 && b > 0) {
    if (b % 2 == c % 2) {
      if (b > c) b = b / 2;
      else c = c / 2;
      printf("*  b = %d c = %d\n", b, c);
    } else {
      b = b / 2;  c = c / 2;
      printf("-  b = %d c = %d\n", b, c);
    }
  }
  if (b > 0 || c > 0)
    printf("|| %d\n", b*10 + c);
  else
    printf("00 %d|\n", 0);
  return 0;
}
```

nusp	a	b	c	saída

2 Questão 2 (2 pontos):

Escreva um programa em C que leia da entrada um número inteiro $n \geq 2$ e que imprima na saída o maior primo que divide n. Por exemplo, para entrada 31, seu programa deve informar que o maior primo que divide 31 é 31. Para entrada 32, ele deve informar que o maior primo que o divide é 2.

3 Questão 3 (3 pontos):

No planeta terra plana da galáxia de centaurus, os números não são representados em base 10 mas em base 100. Não são usados apenas os 10 dígitos binários mas 100 dígitos centenários. Seguindo o costume centenário, um ano possui 100 meses e um mês possui 100 dias. Como em nossa terra, em terra plana um século possui 100 anos. Contudo, o século centenário possui um milhão de dias, e em terra plana todo número 0 a 999999 é chamado de *chato*. Um número chato possui uma *representação* em base 100, em base centenária, *formada por três dígitos centenários amd*, onde *a é o ano*, *m é o mês*, e *d é o dia*. Por exemplo, para o número chato 112233, o ano a é 11, o mês m é 22 e o dia é o dígito centenário menos significativo, 33. Esta representação amd é chamada de *data*, e dizemos que uma data é *feliz* se seus dígitos são todos iguais. Assim, a data do número chato 112233 não é feliz, a de 55555 também não, mas a de 10101 é. Escreva em C um programa que lê do dispositivo de entrada uma sequência de n números chatos e que verifica se a data de cada um é uma data feliz.

4 Questão 4 (3 pontos):

Escreva um programa em C que verifica equipes trapaceiras em um rali de regularidade. O programa recebe como entrada um número inteiro E ($2 < E < 100$) que representa a quantidade de equipes participantes e um número inteiro C ($1 < C < 100$) que representa a quantidade de checkpoints do rali. Em seguida, o programa recebe no máximo $E \times C$ instantes de tempo no formato horas:minutos:segundos sendo que os valores de horas, minutos e segundos são números inteiros respeitando o intervalo de horas de 0 a 99, de minutos de 0 a 59 e de segundos também de 0 a 59. Assim que uma equipe é detectada como trapaceira, os seus valores de entrada não precisam mais ser digitados, o que eventualmente fará essa equipe ter menos do que C tempos informados. É possível ainda que uma equipe tenha menos do que C tempos informados caso ela não tenha conseguido chegar em algum checkpoint e nesse caso o último valor informado para ela terá sido -1:-1:-1.

Uma equipe é considerada trapaceira se o intervalo de tempo entre dois checkpoints consecutivos for menor do que 30 minutos (Observe que equipes que passem pelos checkpoints fora de ordem se enquadram nessa situação pois o intervalo de tempo será negativo). No caso do primeiro checkpoint, considere que o tempo anterior foi 00:00:00 para calcular o intervalo.

A saída do programa deve ser o texto `equipe trapaceira!` assim que for detectada a trapaça. Além disso, depois que o usuário digitar todos os valores de entrada, o programa precisa imprimir a string `quantidade de equipes trapaceiras:` seguido da quantidade de equipes trapaceiras.

Seguem três exemplos de execução do programa. Observe que as instruções para o usuário informar os valores também precisam ser impressas pelo seu programa:

```
digite a quantidade de equipes: 3
digite a quantidade de checkpoints: 2
digite os tempos de passagem nos checkpoints da equipe 0:
-1:-1:-1
digite os tempos de passagem nos checkpoints da equipe 1:
00:29:10
equipe trapaceira!
digite os tempos de passagem nos checkpoints da equipe 2:
00:50:10
01:51:27
quantidade de equipes trapaceiras: 1
```

```
digite a quantidade de equipes: 4
digite a quantidade de checkpoints: 3
digite os tempos de passagem nos checkpoints da equipe 0:
01:00:10
02:02:02
03:03:00
digite os tempos de passagem nos checkpoints da equipe 1:
01:01:01
00:50:10
equipe trapaceira!
digite os tempos de passagem nos checkpoints da equipe 2:
01:01:01
-1:-1:-1
digite os tempos de passagem nos checkpoints da equipe 3:
00:50:50
01:01:10
equipe trapaceira!
quantidade de equipes trapaceiras: 2
```

```
digite a quantidade de equipes: 3
digite a quantidade de checkpoints: 2
digite os tempos de passagem nos checkpoints da equipe 0:
00:50:00
01:50:50
digite os tempos de passagem nos checkpoints da equipe 1:
00:59:10
01:45:00
digite os tempos de passagem nos checkpoints da equipe 2:
01:10:40
02:10:50
quantidade de equipes trapaceiras: 0
```