

Revisão de C

Q1 - Escreva um programa que, infinitamente, leia um número inteiro n , que representa o tamanho, seguida de uma sequência de n números inteiros, e determine o comprimento máximo de um segmento crescente destes n números. Exemplos: Na sequência 5, 10, 3, 2, 4, 7, 9, 8, 5 o comprimento do segmento crescente máximo é 4. Na sequência 10, 8, 7, 5, 2 o comprimento de um segmento crescente máximo é 1. Seu programa para quando n for menor ou igual a zero.

```
#include <stdio.h>

int main() {
    int n;
    while (1) {
        scanf("%d", &n);
        if (n <= 0) break;

        int arr[n];
        for (int i = 0; i < n; i++) scanf("%d", &arr[i]);

        int max_len = 1, curr_len = 1;
        for (int i = 1; i < n; i++) {
            if (arr[i] > arr[i - 1]) {
                curr_len++;
                if (curr_len > max_len) max_len = curr_len;
            } else {
                curr_len = 1;
            }
        }
        printf("Comprimento máximo: %d\n", max_len);
    }
    return 0;
}
```

Q2 - Implemente um programa que, infinitamente, receba, como parâmetro de entrada, um número n e retorne a representação binária de n . Por exemplo, se n é igual a 12, a resposta deste programa deve ser "1100". Seu programa para quando n for menor que zero.

```
#include <stdio.h>

void printBinary(int n) {
    if (n == 0) return;
    printBinary(n / 2);
    printf("%d", n % 2);
}

int main() {
```

```

int n;
while (1) {
    scanf("%d", &n);
    if (n < 0) break;
    if (n == 0) printf("0");
    else printBinary(n);
    printf("\n");
}
return 0;
}

```

Q3- Seja a seguinte seqüência de instruções em um programa C:

```

int *pti;
int veti[]={10,7,2,6,3};
pti = veti;

```

Qual afirmativa é **falsa**?

- a. *pti é igual a 10
- b. *(pti+2) é igual a 2
- c. pti[4] é igual a 3
- d. pti[1] é igual a 10
- e. *(veti+3) é igual a 6

Vamos analisar as alternativas:






- a. *pti == 10 → ☒
- b. *(pti+2) == 2 → ☒
- c. pti[4] == 3 → ☒
- d. pti[1] == 10 → ☒ (pti[1] == 7)
- e. *(veti+3) == 6 → ☒

Resposta correta: d. pti[1] é igual a 10 (**falsa**)

Q4 - Se i e j são variáveis inteiras e pi e pj são ponteiros para inteiro, qual atribuição é ilegal?

- a. pi = &i;
- b. *pj = &j;
- c. *pj = j;
- d. *pi = *pj;

e. `i = (*pi)+*pj;`

- a. ``pi = &i;`` →  válido
- b. ``*pj = &j;`` →  inválido (tentando atribuir endereço a um inteiro)
- c. ``*pj = j;`` →  válido
- d. ``*pi = *pj;`` →  válido
- e. ``i = (*pi) + *pj;`` →  válido

Resposta correta: b. `*pj = &j;` (ilegal)

Listas, Filas e Pilhas

5) Escreva uma função que receba duas Listas (L1 e L2), intercale-as gerando uma terceira Lista, L3

```
Lista* intercalar(Lista* L1, Lista* L2) {
    Lista* L3 = NULL;
    Lista** ptr = &L3;

    while (L1 || L2) {
        if (L1) {
            *ptr = criarNo(L1->valor);
            ptr = &((*ptr)->prox);
            L1 = L1->prox;
        }
        if (L2) {
            *ptr = criarNo(L2->valor);
            ptr = &((*ptr)->prox);
            L2 = L2->prox;
        }
    }
    return L3;
}
```

5.1) Escreva uma função que inverte L1, colocando o resultado em L2 (use uma pilha como auxiliar).

```
void inverterLista(Lista* L1, Lista** L2) {
    Pilha* p = criarPilha();

    while (L1) {
        push(p, L1->valor);
    }
```

```
    L1 = L1->prox;
}

while (!pilhaVazia(p)) {
    int val = pop(p);
    inserirFim(L2, val);
}
}
```