

Número: _____ Nome: _____

1. Implemente a função `int nesimo(int a[], int N, int i)` que dado um array de tamanho $N > 0$ e um inteiro $0 < i \leq N$ devolve o i -ésimo menor elemento do array. Por exemplo, se $i == 1$ a função deve retornar o menor elemento do array.

Número: _____ Nome: _____

2. Implemente a função `LInt removeMaiores(LInt l, int x)` que remove **de uma lista ordenada** `l` todos os elementos maiores que `x`, devolvendo a lista resultante. Considere a definição usual do tipo `LInt`.

```
typedef struct LInt_nodo {  
    int valor;  
    struct LInt_nodo *prox;  
} *LInt;
```

Número: _____ Nome: _____

3. Implemente a função `LInt caminho(ABin a, int x)` que, dada uma **árvore binária de procura** `a` e um valor `x`, devolve uma lista com todos os valores desde a raiz até `x` (inclusivé). Se `x` não existir na árvore, deve devolver `NULL`. Considere a definição usual do tipo `ABin` (o tipo `LInt` foi dado na questão anterior).

```
typedef struct ABin_nodo {  
    int valor;  
    struct ABin_nodo *esq, *dir;  
} *ABin;
```

Número: _____ Nome: _____

4. Implemente a função `void inc(char s[])` que, dada uma string `s` com um número em decimal, incrementa esse número numa unidade. Assuma que a string tem espaço suficiente para armazenar o número resultante. Por exemplo, se a string for "123" deverá ser modificada para "124". Se for "199" deverá ser modificada para "200".

Número: _____ Nome: _____

5. Implemente a função `int sacos(int p[], int N, int C)` que, dado um array com os pesos de N produtos que se pretende comprar num supermercado, e a capacidade C dos sacos desse supermercado, determine o número mínimo de sacos necessários para transportar todos os produtos. Por exemplo, se os pesos dos produtos forem $\{3, 6, 2, 1, 5, 7, 2, 4, 1\}$ e $C == 10$, então bastam 4 sacos. Se os pesos forem $\{3, 3, 3, 3, 5, 5, 11\}$ e $C == 11$, então bastam 3 sacos. Em geral, para descobrir este mínimo teria que testar todas as possíveis maneiras de ensacar os produtos. Se não conseguir implementar essa estratégia ótima, implemente outra que devolva uma aproximação razoável.