Problema A

Considere a seguinte função buscaMenorElemento() que encontra qual é o menor elemento em um vetor:

```
int buscaMenorElemento(int *vetor, int n){
    if(n <= 0) return -1;

    int menor = vetor[0];

    for(int i = 0; i < n; ++i){
        for(int j = i; j < n; ++j){
            if(vetor[i] < menor){
                menor = vetor[i];
            }
        }
    }

    return menor;
}
```

Observe, porém, que essa solução, no pior caso, tem complexidade assintótica igual a O(n²) no número de comparações. Isso porque o *for* interno percorre de i até n-1, mesmo que não seja necessário.

Por exemplo, se n = 5, as iterações são:

Valor de i	# de comparações for interno
0	5
1	4
2	3
3	2
4	1

Ou seja, no total temos o somatório:

$$5 + 4 + 3 + 2 + 1 = \frac{n \cdot (n+1)}{2}$$

que é O(n²).

Entretanto, existe uma forma mais eficiente de fazer este código, de forma que tenha complexidade linear O(n) ao invés de quadrática.

Sua tarefa é implementar a função buscaMenorElemento() de forma mais eficiente (linear) e enviar a função com a seguinte main:

```c
int main() {
    int n;
    scanf("%d", &n);

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

    printf("Menor elemento: %d\n", buscaMenorElemento(v, n));

    return 0;
}
```

Entrada

A primeira linha possui o número de elementos do vetor. A segunda linha possui os elementos separados por espaço.

Saída

Imprimir “Menor elemento: “, seguido do menor elemento no vetor.

Exemplo de entrada	Exemplo de saída
4 1 55 6 3	Menor elemento: 1

Exemplo de entrada	Exemplo de saída
4 99 99 99 5	Menor elemento: 5