Problema B

Considere a seguinte função abaixo que verifica se uma string é palíndromo. A função retorna 1 se a string for palíndromo ou 0, caso contrário.

```
int contaPares(int *vetor, int n){
    int contador = 0;

    for(int i = 0; i < n; ++i){
        for(int j = 0; j < n; ++j){
            if(i == j){
                if(vetor[i] % 2 == 0){
                    contador += 1;
                }
            }
        }
    }

    return contador;
}
```

Observe, porém, que essa solução, no pior caso, tem complexidade assintótica igual a $O(n^2)$ no número de comparações.

Entretanto, existe uma forma mais eficiente de fazer este código, com uma complexidade menor que quadrática. Assim, sua tarefa é implementar a função *contaPares()* de forma mais eficiente e enviar a função com a seguinte main:

```
// Complexidade = O(___)
int main(){
    int n;
    scanf("%d", &n);

    int v[n];

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

    printf("%d\n", contaPares(v, n));

    return 0;
}
```

Você deverá preencher a complexidade da *contaPares()* que você produziu no comentário antes da main. Caso o código esteja correto mas a complexidade não, você receberá o código “Name mismatch”!

Entrada
A primeira linha possui um inteiro n, a quantidade de elementos do vetor. A próxima linha contém n inteiros, os elementos do vetor.

Saída
A quantidade de elementos pares no vetor.

Exemplo de entrada	Exemplo de saída
5 1 2 3 4 5	2

Exemplo de entrada	Exemplo de saída
4 2 2 2 2	4