Universidade Federal de Uberlândia

Profa. Christiane Regina Soares Brasil

Email: christiane.ufu@gmail.com

Aula prática - 07/10/2022

1) Calcule o número de instruções em função de N nos trechos de códigos apresentados abaixo, considerando o melhor e o pior caso (determine-os).

a)

```
...
int i;
for(i = 0; i<n; i++)
{
        printf("%d ", A[i]);
}
```

*F(n) = 3n + 2 → **para todos os casos (Caso Geral)***

b)

```
...
int i;
for( i = N-1; i >= 0; i--)
{
    A[i+1] = A[i];
}
A[0] = x;
```

1 instrução para i = N-1; // executa a primeira vez
1 instrução para i>=0; //executa a primeira vez
n instruções para A[i+1] = A[i]; //executa n vezes
n instruções para i--; //executa n vezes
n instruções para i>=0; //executa n vezes
1 instrução para A[0] = A[x]; // executa 1 vez

F(n) = 1 + 1 + n + n + n + 1 = 3n + 3
*F(n) = 3n + 3 → **para todos os casos (Caso Geral)***

c)

```
...
```

```
int i;
for(i = 0; i<n; i++)
{
        if(A[i]%2==0) printf("%d ", A[i]);
}
```

Melhor caso (todos são ímpares):

1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(A[i]%2==0) ; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para i<n; //executa n vezes

F(n) = 1 + 1 + n + n + n = 3n + 2
F(n) = 3n + 2

Pior caso (todos são pares):

1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(A[i]%2==0) ; //executa n vezes
n instruções para printf("%d ", A[i]); //executa n vezes
n instruções para i++; //executa n vezes
n instruções para i<n; //executa n vezes

F(n) = 1 + 1 + n + n + n + n = 4n + 2
F(n) = 4n + 2

d)

```
...
int i, cont = 0;
float media = 0;
for(i = 0; i<n; i++)
{
        if(A[i]%2==0) { media += A[i]; cont++; }
}
if(cont==0) printf(“Nenhum par.”);
else    media = media/cont;
```

Melhor caso (todos são ímpares):

1 instrução para cont = 0; // executa 1 vez
1 instrução para media = 0; // executa 1 vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(A[i]%2==0) ; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para i<n; //executa n vezes
1 instrução para if(cont==0); // executa 1 vez
1 instrução para printf("Nenhum par"); // executa 1 vez

F(n) = 1 + 1 + 1 + 1 + n + n + n + 1 + 1 = 3n + 6
F(n) = 3n + 6

Pior caso (todos são pares):
1 instrução para cont = 0; // executa 1 vez
1 instrução para media = 0; // executa 1 vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(A[i]%2==0) ; //executa n vezes
n instruções para media += A[i]; //executa n vezes
n instruções para cont++; //executa n vezes
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
1 instrução para if(cont==0); // executa 1 vez
1 instrução para media = media/cont;// executa 1 vez

F(n) = 1 + 1 + 1 + 1 + n + n + n + n + n + 1 + 1 = 5n + 6
F(n) = 5n + 6

e)
```
...
 int i, max;
max = A [0];
for(i = 0; i<n; i++)
{
        if(max < A[ i ])  max = A[ i ];
}
```

Melhor caso (o maior elemento é A[0]):

1 instrução para max = A[0]; // executa 1 vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(max < A[i]) ; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para i<n; //executa n vezes

F(n) = 1 + 1 + 1 + n + n + n = 3n + 3
F(n) = 3n + 3

Pior caso (o vetor está ordenado de modo crescente):
1 instrução para max = A[0]; // executa 1 vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para if(max < A[i]) ; //executa n vezes
n instruções para max = A[i] ; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para i<n; //executa n vezes

F(n) = 1 + 1 + 1 + n + n + n + n = 4n + 3
F(n) = 4n + 3

f)
```
...
int i, j;
for(i = 0; i<n; i++)
      for(j = 0; j<n; j++)
          printf("%d", A[i][j]);
```

1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para j = 0; //executa n vezes
n instruções para j<n; // executa n vezes
*n^2 instruções para j<n; //executa n*n vezes*
*n^2 instruções para j++; //executa n*n vezes*
*n^2 instruções para printf("%d", A[i][j]); //executa n*n vezes*
F(n) = 1 + 1 + n + n + n + n + n^2 + n^2 + n^2 = $3n^2$ + 4n + 2

*$F(n) = 3n^2 + 4n + 2$ → **para todos os casos (Caso Geral)***

g) ...

```
int i, j;
    for(i = 0; i<n; i++)
        for(j = 0; j<n; j++)
            if( A[ i ][ j ]==0)  A[ i ][ j ]=1;
```

Melhor caso (não há zero na matriz):

1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para j = 0; //executa n vezes
n instruções para j<n; // executa n vezes
*n^2 instruções para j<n; //executa n*n vezes*
*n^2 instruções para j++; //executa n*n vezes*
*n^2 instruções para if(A[i][j]==0); //executa n*n vezes*

$F(n) = 1 + 1 + n + n + n + n + n^2 + n^2 + n^2 = 3n^2 + 4n + 2$
$F(n) = 3n^2 + 4n + 2$

Pior caso (todos são zeros na matriz):

1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para j = 0; //executa n vezes
n instruções para j<n; // executa n vezes
*n^2 instruções para j<n; //executa n*n vezes*
*n^2 instruções para j++; //executa n*n vezes*
*n^2 instruções para if(A[i][j]==0); //executa n*n vezes*
*n^2 instruções para A[i][j]=1; //executa n*n vezes*

$F(n) = 1 + 1 + n + n + n + n + n^2 + n^2 + n^2 + n^2 = 4n^2 + 4n + 2$
$F(n) = 4n^2 + 4n + 2$

h) ...

```
int i, j, max;
    max = A [0][0];
    for(i = 0; i<n; i++)
        for(j = 0; j<n; j++)
            if(max < A[ i ][ j ])  max = A[ i ][ j ];
```

Melhor caso (*o maior elemento é A[0][0]*):

1 instrução para max = A[0][0]; // executa a primeira vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para j = 0; //executa n vezes
n instruções para j<n; // executa n vezes
*n^2 instruções para j<n; //executa n*n vezes*
*n^2 instruções para j++; //executa n*n vezes*
*n^2 instruções para if(max < A[i][j]); //executa n*n vezes*

$F(n) = 1 + 1 + 1 + n + n + n + n + n^2 + n^2 + n^2 = 3n^2 + 4n + 3$
$F(n) = 3n^2 + 4n + 3$

Pior caso (matriz em ordem crescente por linha):

1 instrução para max = A[0][0]; // executa a primeira vez
1 instrução para i = 0; // executa a primeira vez
1 instrução para i<n; //executa a primeira vez
n instruções para i<n; //executa n vezes
n instruções para i++; //executa n vezes
n instruções para j = 0; //executa n vezes
n instruções para j<n; // executa n vezes
*n^2 instruções para j<n; //executa n*n vezes*
*n^2 instruções para j++; //executa n*n vezes*
*n^2 instruções para if(max < A[i][j]); //executa n*n vezes*
*n^2 instruções para max = A[i][j]; //executa n*n vezes*

$F(n) = 1 + 1 + 1 + n + n + n + n + n^2 + n^2 + n^2 + n^2 = 4n^2 + 4n + 3$
$F(n) = 4n^2 + 4n + 3$