

# Sottoarray di somma massima

Sezione 2.1 [CGG]

# Redirezione dell'input

Meccanismo per prendere dati in input direttamente da file di testo. Questa tecnica sarà utilizzata per la correzione automatica delle prove di laboratorio.

Sintassi Tipica: `prog < input`

`./esercizio.o < input`

# Sottoarray di Somma Massima

Problema: dato un array di  $n$  interi (anche negativi) individuare la sottoarray di somma massima.

Input: un array di interi (anche negativi)

Output: valore della somma

Esempio:

Input    -1 5 8 -9 1 1



Output            13

# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```

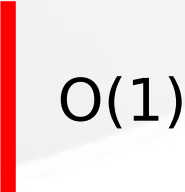
# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```

Input  -1 5 8  -9 1 1

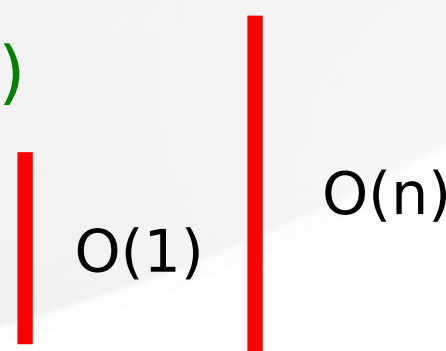
# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```

 O(1)

# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```



The diagram shows two vertical red lines. The shorter line on the left is positioned between the innermost curly braces of the innermost loop and the line `somma+=a[k];`. The label `O(1)` is placed to the right of this line. The taller line on the right is positioned between the curly braces of the middle loop and the line `if(somma > max) max=somma;`. The label `O(n)` is placed to the right of this line.

# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```

The diagram illustrates the time complexity of the code blocks. Three red vertical bars are positioned to the right of the code. The shortest bar, labeled  $O(1)$ , is aligned with the innermost loop body (the `somma+=a[k];` line). The medium-height bar, labeled  $O(n)$ , is aligned with the middle loop (the `for(k=i; k<=j; k++)` line). The tallest bar, labeled  $O(n^2)$ , is aligned with the outermost loop (the `for(j=i; j<n; j++)` line).



# Soluzione 1

```
max=a[0];
for(i=0; i<n; i++)
{
    for(j=i; j<n; j++)
    {
        somma=0;
        for(k=i; k<=j; k++)
        {
            somma+=a[k];
        }
        if(somma > max) max=somma;
    }
}
```

O(1)

O(n)



O(n<sup>2</sup>)

O(n<sup>3</sup>)

Tempo: O(n<sup>3</sup>) :-)

# Soluzione 2

```
max=a[0];
for(i=0; i<n; i++)
{
    somma=0;
    for(j=i; j<n; j++)
    {
        somma+=a[j];
        if(somma > max) max=somma;
    }
}
```

Input     -1   5   8    -9   1   1

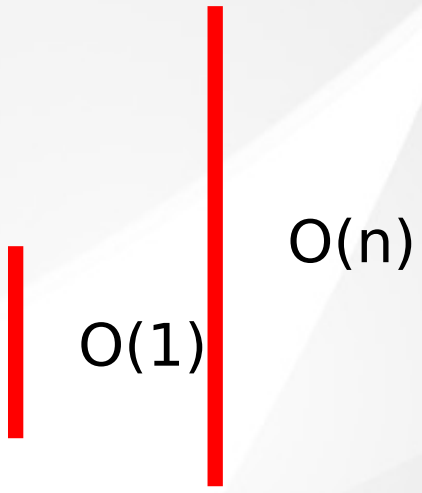
# Soluzione 2

```
max=a[0];
for(i=0; i<n; i++)
{
    somma=0;
    for(j=i; j<n; j++)
    {
        somma+=a[j];
        if(somma > max) max=somma;
    }
}
```

**O(1)**

# Soluzione 2

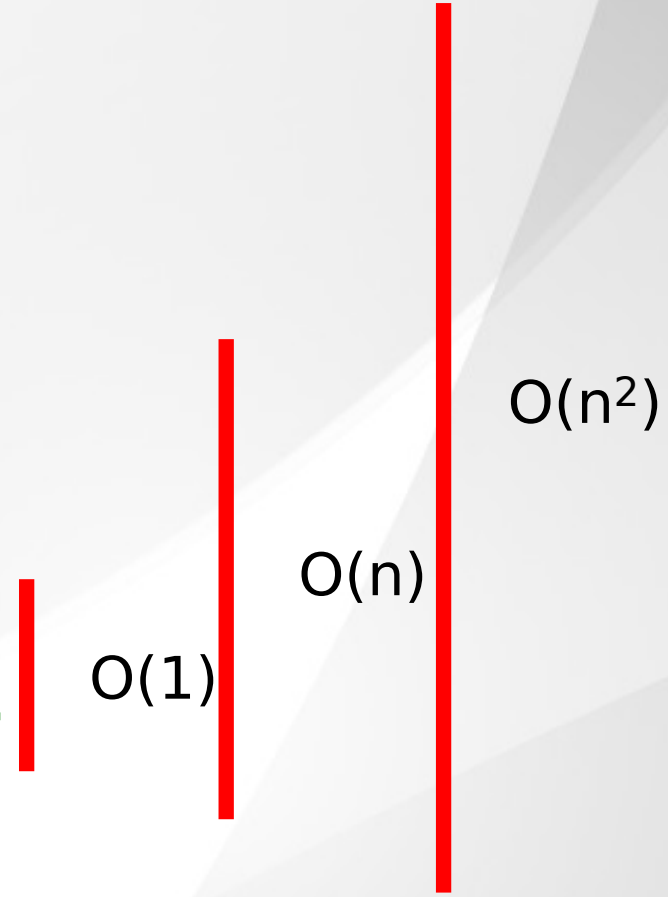
```
max=a[0];
for(i=0; i<n; i++)
{
    somma=0;
    for(j=i; j<n; j++)
    {
        somma+=a[j];
        if(somma > max) max=somma;
    }
}
```



The diagram consists of two vertical red lines. The shorter line on the left is positioned to the right of the innermost loop's closing brace, with the label  $O(1)$  to its right. The taller line on the right is positioned to the right of the outermost loop's closing brace, with the label  $O(n)$  to its right.

# Soluzione 2

```
max=a[0];
for(i=0; i<n; i++)
{
    somma=0;
    for(j=i; j<n; j++)
    {
        somma+=a[j];
        if(somma > max) max=somma;
    }
}
```



Tempo:  $O(n^2)$  :-|

# Come fare meglio?

- Possiamo sfruttare due proprietà del sottoarray di somma massima
  - 1) La somma dei valori in ogni prefisso del **sottoarray ottimo** è positiva, se così non fosse potremmo eliminare tale prefisso ottenendo un sottoarray di somma maggiore (**assurdo**)
  - 2) Il valore immediatamente precedente al primo valore del **sottoarray ottimo** è negativo, se così non fosse potremmo aggiungere tale valore ottenendo un sottoarray di somma maggiore (**assurdo**)

-1 5 8 -9 1 1

# Soluzione 3

```
max = A[0]; somma = 0;
for(i=0; i<n; i++)
{
    if(somma > 0) somma+=a[i];
    else somma=a[i];

    if(somma > max) max=somma;
}
```

Tempo:  $O(n)$  :-)

# Soluzione 3

```
max = A[0]; somma = 0;
for(i=0; i<n; i++)
{
    if(somma > 0) somma+=a[i];
    else somma=a[i];

    if(somma > max) max=somma;
}
```

Algoritmo ottimo!

Perché?

Tempo:  $O(n)$  :-)



# Esempio soluzione lineare

**1** 2 -4 1 3 2 -2 1

SOMMA    MAX  
1            1

# Esempio soluzione lineare

								SOMMA	MAX
1	2	-4	1	3	2	-2	1	1	1
1	2	-4	1	3	2	-2	1	3	3

# Esempio soluzione lineare

	SOMMA	MAX
1 2 -4 1 3 2 -2 1	1	1
1 2 -4 1 3 2 -2 1	3	3
1 2 -4 1 3 2 -2 1	-1	3

# Esempio soluzione lineare

	SOMMA	MAX
1 2 -4 1 3 2 -2 1	1	1
1 2 -4 1 3 2 -2 1	3	3
1 2 -4 1 3 2 -2 1	-1	3
1 2 -4 1 3 2 -2 1	1	3

# Esempio soluzione lineare

	SOMMA	MAX
1 2 -4 1 3 2 -2 1	1	1
1 2 -4 1 3 2 -2 1	3	3
1 2 -4 1 3 2 -2 1	-1	3
1 2 -4 1 3 2 -2 1	1	3
1 2 -4 1 3 2 -2 1	4	4
1 2 -4 1 3 2 -2 1	6	6
1 2 -4 1 3 2 -2 1	4	6
1 2 -4 1 3 2 -2 1	5	6

# Esempio soluzione lineare

	SOMMA	MAX
1 2 -4 1 3 2 -2 1	1	1
1 2 -4 1 3 2 -2 1	3	3
1 2 -4 1 3 2 -2 1	-1	3
1 2 -4 1 3 2 -2 1	1	3
1 2 -4 1 3 2 -2 1	4	4
1 2 -4 1 3 2 -2 1	6	6
1 2 -4 1 3 2 -2 1	4	6
1 2 -4 1 3 2 -2 1	5	6

