

# 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`

**Importante:** durante le prove di laboratorio attenersi strettamente alle specifiche sul formato dell'output.

# Sottoarray di Somma Massima

Problema: data 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;
    }
}
```

# 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

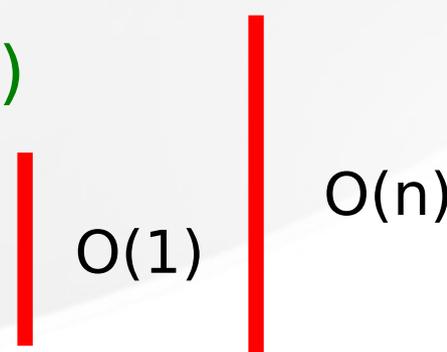
# Soluzione 1

```
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;
    }
}
```

A vertical red line is positioned to the right of the innermost loop's closing brace, with the text  $O(1)$  to its right, indicating the time complexity of that specific operation.

# 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 labeled  $O(1)$  and is positioned to the right of the innermost loop's closing brace. The taller line on the right is labeled  $O(n)$  and is positioned to the right of the middle loop's closing brace.

# 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 vertical red bars are positioned to the right of the code. The first bar, located under the innermost loop, is labeled  $O(1)$ . The second bar, located under the middle loop, is labeled  $O(n)$ . The third bar, located under the outermost loop, is labeled  $O(n^2)$ .

# 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;
    }
}
```

$O(1)$

$O(n)$

$O(n^2)$

$O(n^3)$

Tempo:  $O(n^3)$  :-)

# 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

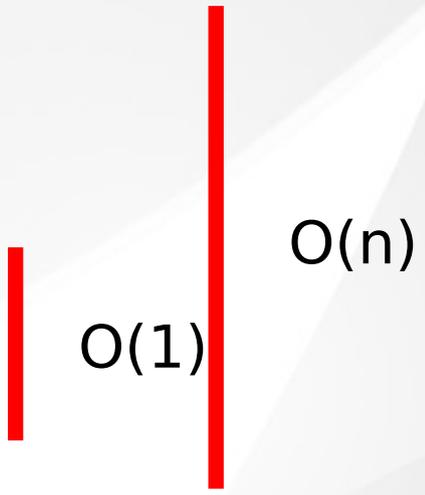
# Soluzione 2

```
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

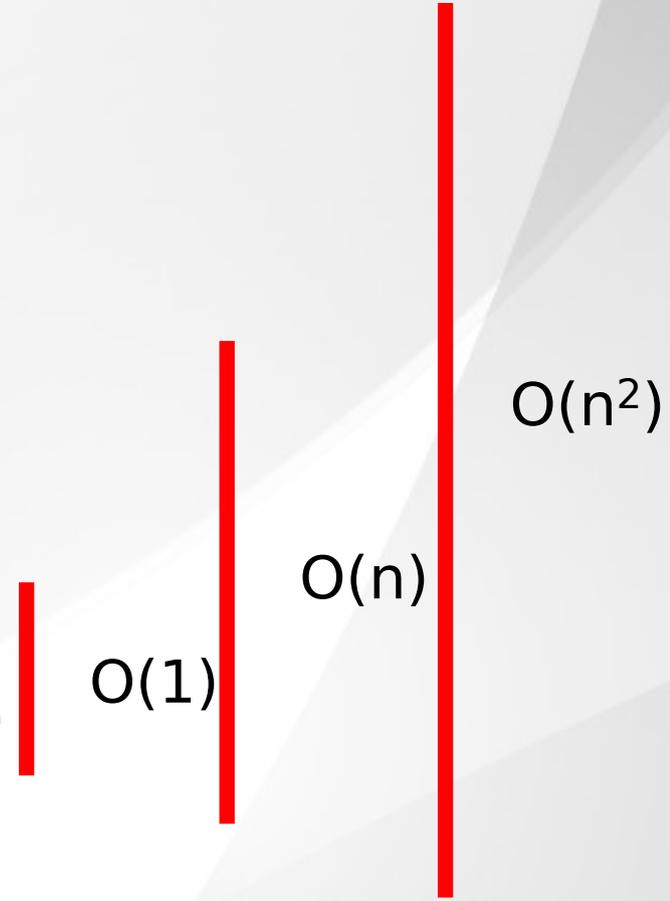
```
max=a[0];
for(i=0; i<n; i++)
{
    somma=0;
    for(j=i; j<n; j++)
    {
        somma+=a[j];
        if(somma > max) max=somma;
    }
}
```



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];  
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];  
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

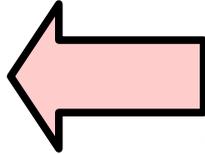
	SOMMA	MAX
1 2 -4 1 3 2 -2 1	1	1
1 2 -4 1 3 2 -2 1	3	3
1 2 -4 1 3 2 -2 1	-1	3
1 2 -4 1 3 2 -2 1	1	3

# 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



# Esercizio 1

Implementare le tre soluzioni per il problema del “sottoarray di somma massima”. Per ciascuna di esse scrivere un programma che legga la sequenza di input da terminale e stampa in output il valore massimo della somma del sottoarray di somma massima.

Si assuma che la prima riga dell'input contenga la lunghezza della sequenza

```
INPUT    3          OUTPUT  4
         1
        -2
         4
```

Provare le soluzioni sui file di input che trovate sul sito del corso, utilizzando la redirectione dell'input.

**Linux:** potete usare il comando `time` per ottenere i tempi di esecuzione del vostro programma (utile per confrontare le prestazioni).

```
time ./maxsum.o < input
```

# Esercizio 2

**Intersezione tra insiemi:** Scrivere un programma che accetti in input dall'utente due sequenze di interi distinti e stampi gli elementi che occorrono in entrambi. Si assume che l'utente specifichi la lunghezza di ogni sequenza prima dell'immissione degli elementi.