Największa suma spójnego fragmentu – przykłady rozwiązań

Dane: zbiór A zawierający N liczb całkowitych

Metoda 1 – O(n^3)

Dla każdej pary indeksów i oraz j takich, że 1≤i≤j≤N obliczamy sumę elementów A[i]..A[j] i sprawdzamy czy ona jest największa z dotychczasowych sum:

```
max:=A[1];
for i:=1 to N do
 for j:=i to N do
begin
  suma:=0;
  for k:=i to j do
          suma:=suma+A[k];
  if suma>max then max:=suma;
end;
```

Metoda 2 – O(n^2)

Dla kolejnych i będących pozycją początku podciągu, obliczamy sumę podciągu kolejno wydłużanego o element z pozycji j.

```
max:= A[1];
for i:=1 to N do
begin
  suma:=0;
  for j:=i to N do
        begin
          suma:=suma+A[i];
          if suma>max then max:=suma;
          end;
end;
```

Metoda 3 – $O(n^2)$

Tworzymy tablicę suma, która zawiera sumę częściową elementów od 1 do i. Sumę elementów A[k,m] obliczamy: suma[m]-suma[k-1]

```
suma[0]:=0;
for i:=1 to N do
  suma[i]:=suma[i-1]+A[i]
max:=A[1];
for k:=1 to N do
 for m:=k to N do
 begin
     suma:=suma[m]-suma[k-1];
     if suma>max then max:=suma
 end;
```

Metoda 4 – O(n)

Kolejny element (z pozycji i) albo wydłuża podciąg mający szansę być podciągiem o największej sumie, albo go kończy. Gdy suma dotychchczasowego podciągu i i-tego elementu jest większa od 0 to obliczana jest suma. Gdy zaś ta suma jest mniejsza od 0 to oznacza, że należy rozpocząć rozpatrywanie nowego podciągu z początkiem w pozycji i+1.
Jeśli w trakcie badania ciągu okaże się, że maksymalną sumą jest 0 to oznacza, że ciąg zawiera tylko liczby niedodatnie. W takim przypadku algorytm wyszukuje największą z liczb ciągu.

```
max:=0;
maxkon:=0;
for i:=1 to N do
begin
  maxkon:=maksymalna(maxkon+A[i],0)
  max:= maksymalna(max, maskon)
end;
if max=0 then
 begin
     max:=A[1];
     for i:=2 to N do
         if A[i]>max then max:=A[i]
 end;
```