1. 1/0 Knapsack problem – algoritmul pseudo-polinomial

Input:  
W <- int;

A,B <-vectori de lungime n;  
  
Output:  
Un vector X de lungime n format din 1 si 0, astfel incat

& – maxim

Programare dinamica:  
T[i][g] = profitul maxim obtinut selectand elemente din intervalul [1,…,i] fara a depasi greutatea totala g

T[n][W]-> va fi solutia noastra;

T[0][g]=T[i][0]=0

T[i][g]=max(T[i-1][g], val(i)+T[i-1][g-weight(i)]);

Complexitate timp: O(n\*W)  
Spatiu O(W)

n=3

A={60, **100, 120**}; B={10,**20,30**}; W=50;  
  
  
  
  
  
Fie urmatorul joc:

Se de un vector W de numare positive

w1, w2, w3, ….., wn

Fiecare dintre cei doi jucatori poate alege unul dintre capetele vectorului, eliminand-ul din vector si insusindu-si valoarea elementului ca punctaj obtinut. Jocul se desfasoara prin succesiune de mutari din partea ambilor jucatori.   
Jocul se termina cand se epuizeaza elementele din vector.

Sa se scrie un program care “ajuta” pe primul jucator sa faca mutarea optima, si la fiecare pas sa ii spuna jucatorului care este punctajul minim garantat pe care il va putea obtine pana la final.

W[i……j]

T[i][j] – punctajul maxim garantat pe care il poate obtine jucatorul avand in fata table de joc W[i:j]

T[1][n]

T[i][i]= wi

T[i][i+1]= max (wi, w(i+1))

For (i=0;i<n;i++)

For(len=2;i+len<n;len++)

T[i][i+len]=max(w[i]+min(T[i+2][i+len],T[i+1][i+len-1]),…..)

**wi** w(i+1), w(i+2) …… w(j-2), w(j-1), *wj*