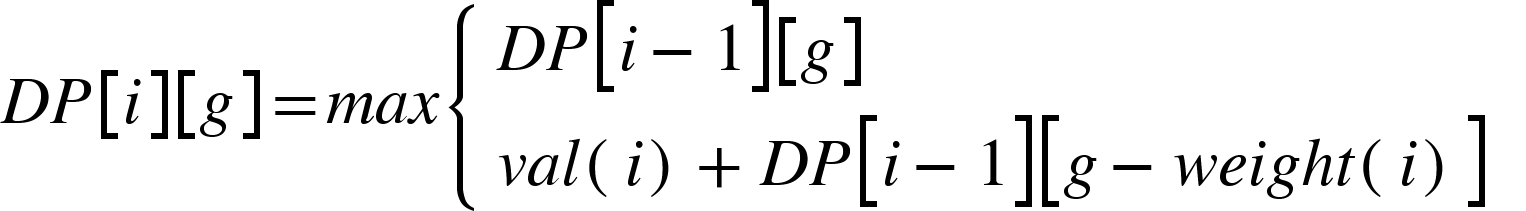
1/0 Knapsack problem

* problema rucsacului in varianta in care obiectele nu pot fi taiate

Programare dinamica

DP[i][g] = profitul maxim ce poate fi obtinut selectand dintre primele *i* obiecte, fara a depasi greutatea totala *g*

DP[0][g]=DP[i][0]=0;



Scrieti un program care citeste dintr-un fisier datele de intrare si returneaza valoarea maxima obtinuta, precum si obiectele selectate:  
INPUT:

W=50 - capacitatea rucsacului

val=[60,100,120]

w=[10,20,30]

OUTPUT:  
220; [0,1,1];

complexitate:

O(n\*W) - complexitate pseudo-polinomiala deoarece capacitatile sunt intregi

2) Joc

avem 2 jucatori si un sir S. Jocul se desfasoara sub unor succesiuni de mutari. La o mutare, jucatorul curent alege unul dintre capetele lui S, “rupe “din sir elementul din acel capat. Aduna vaoloarea elementului la punctajul sau. Jocul se termina atunci cand S devine vid. Jucatorul cu punctajul cel mai mare este declarat castigator.

1. Fie S de lungime para. Dari exemplu de o tactic simpla ce poate fi abordata de catre jucatorul 1, astfel incat acesta sa obtina mereu cel putin remiza.

Jucatorul poate face suma elementelor de pe pozitie para, respectiv impara si apoi poate forta jocul astfel incat el sa ia mereu elemente de aceiasi paritate.

1. dati un algoritm care sa maximizeze punctajul jucatorului aflat la mutare

[1…...n]

[i……...j]

T[i][j] - punctajul maxim ce poate fi obtinut din sirul S[i….j]

T[i][i]=S[i]

T[i][i+1]= max(S[i],S[i+1])

