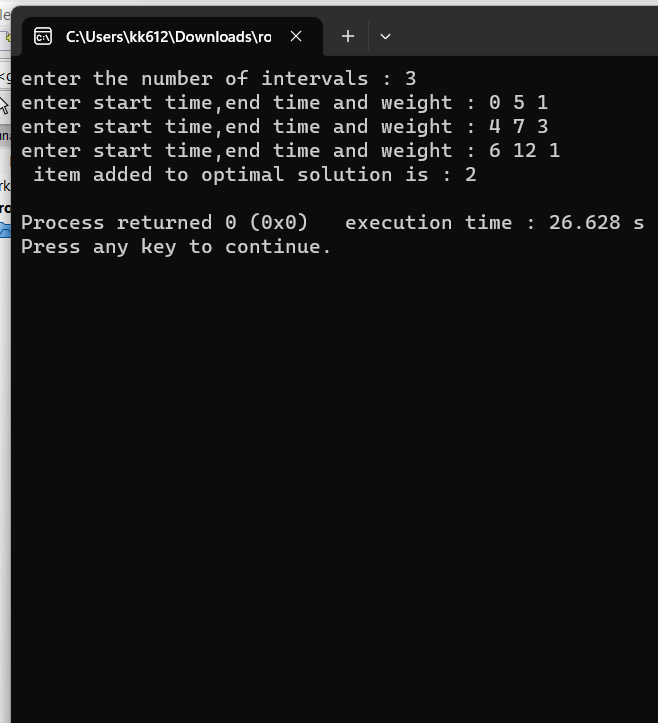
**10.) weighted interval**



#include <iostream>

using namespace std;

class interval{

public:

int weight;

int start;

int finish;

int number;

interval(){

start =0;

finish = 0;

weight = 0;

number = 0;

}

void set\_data(int s,int f,int w,int num){

start =s;

finish = f;

weight = w;

number = num;

}

};

int memoization(int j,int \*m ,int size){

if(j=0){

return 0;

}else if(m[j] != 0){

return m[j];

}else{

}

}

int p(int i, int size,interval \*interval\_list){

for(int j=i-1;j>=0;j--){

if(interval\_list[j].finish <= interval\_list[i].start ){

return j;

}

}

return 0;

}

//complete iteration and memoization

void weighted\_interval\_scheduling(int size,interval \*interval\_list){

int \*m = new int[size+1]; //to save the optimal answers

int \*pp = new int[size+1]; //it stores the rest compatible intervals for every interval

for(int j=1;j<=size;j++){

pp[j] =p(j, size, interval\_list);

m[j] = max(interval\_list[j].weight + m[pp[j]] , m[j-1]);

}

for(int j=size;j>=1;){ //start from last

if( interval\_list[j].weight+ m[pp[j]] >= m[j-1]){

cout<<" item added to optimal solution is : "<<j<<endl;

j = pp[j];

}else{

j--;

}

}

}

//only recursion

int weighted\_interval\_recursive(int size,interval \*interval\_list,int n){

int \*pp = new int[size+1]; //it stores the rest compatible intervals for every interval

for(int j=1;j<=size;j++){

pp[j] =p(j, size, interval\_list);

}

if(n==0){

return 0;

}

return max( interval\_list[n].weight + weighted\_interval\_recursive(size,interval\_list,pp[n]),weighted\_interval\_recursive(size,interval\_list,n-1));

}

//recursion + memoization

int mm[100];

int weighted\_interval\_recursive(int size,interval \*interval\_list,int n, int \*pp){

if(n==0){

return 0;

}

if(mm[n]==-1)

return mm[n] = max( interval\_list[n].weight + weighted\_interval\_recursive(size,interval\_list,pp[n],pp),weighted\_interval\_recursive(size,interval\_list,n-1,pp));

else

return mm[n];

}

int main()

{

int size =0;

cout<<"enter the number of intervals : ";

cin>>size;

interval \*interval\_list = new interval[size+1];

for(int i=1;i<=size;i++){

cout<<"enter start time,end time and finish time : ";

int s=0,f=0,w=0;

cin>>s>>f>>w;

interval\_list[i].set\_data(s,f,w,1);

}

weighted\_interval\_scheduling(size,interval\_list);

/\* int \*pp = new int[size+1]; //it stores the rest compatible intervals for every interval

for(int j=1;j<=size;j++){

pp[j] =p(j, size, interval\_list);

}

for(int i=0;i<50;i++){

mm[i] =-1;

}

cout<<"RESULT : "<<weighted\_interval\_recursive(size,interval\_list,size,pp);

\*/

return 0;

}