**Algorithm**

**201133216 정유석**

***Smart Elephant(Using Dynamic table)***

**CODE:**

#include<stdio.h>

#include<stdlib.h>

#define MAX\_WEIGHT 10000

#define MAX\_ELEPHANT 1000

typedef struct { //Structure is store the information of Elephant

int weight, IQ, number;

}Elephant;

typedef struct{ //It store the Number of Elephant

int num;

}Number;

Elephant ele[MAX\_ELEPHANT],ele\_find[MAX\_ELEPHANT][MAX\_ELEPHANT]; //ele is main information and ele\_find is order of Weight Increase, IQ decrease

Number num[MAX\_ELEPHANT]; //Number of Elephant

void main(){

int cnt,i,j, min\_w, min\_i, min\_n; //cnt is number of elephant, i-j loop value, k is using number of elephant find, min\_w-n-e-i is using find minimum value

int temp\_w, temp\_i, temp\_n, temp\_IQ; //temporary value

int length[MAX\_ELEPHANT]; //Each length

int table[100][100];

int position = 0 ;

int number[100][100],compare\_IQ,compare\_weight;

int max=0;

int t\_weight, index[100], count=0,l,m,k,temp2[100],temp3;

printf("Entert number of ephant number : ");

scanf("%d", &cnt);

while(cnt>MAX\_ELEPHANT || cnt<1){

printf("Again Input the elephant number (1~1000) : ");

scanf("%d",&cnt);

}

printf("Input Elephant Weight and IQ : \n");

for(i =0; i <cnt; i ++){

scanf("%d %d",&ele[i].weight,&ele[i].IQ);

while(ele[i].weight>MAX\_WEIGHT || ele[i].weight <0 || ele[i].IQ > 10000 || ele[i].IQ < 0){

printf("Again INPUT the weight and IQ (1~ 10000) : ");

scanf("%d %d",&ele[i].weight,&ele[i].IQ);

}

ele[i].number = i+1;

}

//-- array is ordered small weight

for(i=0; i<cnt; i++){ //Use the bubble sort

for(j=0; j<cnt-1; j++){

if(ele[j].weight > ele[j+1].weight){ //find the minmum of weight and insert to array order

temp\_w = ele[j+1].weight;

temp\_i = ele[j+1].IQ;

temp\_n = ele[j+1].number;

ele[j+1].weight = ele[j].weight;

ele[j+1].IQ = ele[j].IQ;

ele[j+1].number = ele[j].number;

ele[j].IQ = temp\_i;

ele[j].number = temp\_n;

ele[j].weight = temp\_w;

}

else

continue;

}

}

for(i=0; i<cnt; i++){

position = 0;

number[i][position]=ele[i].number;

compare\_IQ = ele[i].IQ;

compare\_weight = ele[i].weight;

for(j=0; j<cnt; j++){

if(i==j || i>j){

table[i][j] = 0;

continue;

}

else{

if(compare\_IQ>ele[j].IQ && compare\_weight < ele[j].weight){

position++;

table[i][j] = position;

compare\_IQ = ele[j].IQ;

compare\_weight = ele[j].weight;

}

else

table[i][j] = position;

}

}

}

for(i=0; i<cnt;i++){

for(j=0; j<cnt; j++){

printf("%d ",table[i][j]);

if(max<table[i][j])

max = table[i][j];

}

printf("\n");

}

printf("\nMAX longest is %d ! \n",max+1);

for(i=0; i<cnt; i++){ //Part of backtrace

for(j=0; j<cnt; j++){

if(table[i][j] == max){

t\_weight = ele[j].weight;

index[count] = ele[j].number;

count++;

for(k=j-1; k>0; k--){

if(table[i][k] != max && table[i][k] - table[i][k-1] == 1

&& ele[k].weight != ele[k-1].weight){

for(m=0; m<cnt; m++){

if(table[i][m] == table[i][k] && m!=k){

temp2[m] = ele[m].number;

temp3 = count;

}

}

index[count] = ele[k].number;

count++;

}

}

index[count] = ele[i].number;

printf("---------index----------\n"); //Print them

for(l=count; l>=0; l--)

printf("%d\n",index[l]);

for(m=0; m<cnt; m++){

if(temp2[m]>0){

index[temp3] = temp2[m];

printf("---------index----------\n");

for(l=count; l>=0; l--)

printf("%d\n",index[l]);

}

}

count =0;

}

}

}

}

**Print:**



