**알고리즘**

**(Smart Elephant)**

**200932155 우영남**

#include <stdio.h>

#include <stdlib.h>

typedef struct ElepantInfo ElephantInfo;

struct ElepantInfo

{

int Weight;

int IQ;

int index;

};

void insertInfo(ElephantInfo\* info, int numOfInfo);//정보를 입력받는 함수입니다

void sortInfoByIQ(ElephantInfo\* info, int numOfInfo);//정보를 아이큐순으로 솔팅하는 함수입니다

void calculator(ElephantInfo\* info, int numOfInfo, int i, int j, int\* maxWeight,int\*minIQ);//결과값을 계산하는 함수입니다

void backTrace(ElephantInfo\* info, int numOfInfo);//출력을 하는 함수입니다

int\*\* table;

int main(void)

{

int i, j;

int numOfInfo=0;

while(1)

{

printf("Enter the number of INFORMATION(1~1000) : ");

scanf("%d",&numOfInfo);

if(numOfInfo>=1 && numOfInfo<=1000)

{

break;

}

else

{

printf("[ERROR!]number of INFORMATION must between 1 to 1000!\n\n");

}

}

//이중배열 "table" 메모리할당 및 초기화//

table = (int\*\*)malloc(sizeof(int\*)\*(numOfInfo+1));

for(i=0; i<(numOfInfo+1); i++)

{

table[i] = (int\*)malloc(sizeof(int)\*(numOfInfo+1));

}

for(i=0; i<numOfInfo+1; i++)

{

for(j=0; j<numOfInfo+1; j++)

{

if(i==0 || j==0)

{

table[i][j]=0;

}

}

}

//

//정보갯수만큼 ElephantInfo 메모리할당//

ElephantInfo\* info = (ElephantInfo\*)malloc(sizeof(ElephantInfo)\*numOfInfo);

//

//코끼리의 정보를 입력시키는 함수입니다//

insertInfo(info, numOfInfo);

//

//아이큐순으로 정보를 정렬하는 함수입니다//

sortInfoByIQ(info, numOfInfo);

/\*

for(i=0; i<numOfInfo; i++)

{

printf("%d's Weight is %d // %d's IQ is %d\n",i+1, info[i].Weight, i+1, info[i].IQ);

}

\*/

//

//함수 calculator을 돌려 table을 완성하는 과정입니다//

int max=0;

int min=0;

int\* maxWeight=&max;

int\* minIQ=&min;

for(i=0; i<numOfInfo; i++)

{

\*maxWeight = info[i].Weight;

\*minIQ = info[i].IQ;

for(j=0; j<numOfInfo; j++)

{

calculator(info, numOfInfo, i, j, maxWeight,minIQ);

}

}

//

//이중배열 "Table" 출력부분입니다//

for(i=0; i<(numOfInfo+1); i++)

{

for(j=0; j<(numOfInfo+1); j++)

{

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

}

printf("\n");

}

//

//출력을 하는 부분입니다//

backTrace(info, numOfInfo);

//

}

void backTrace(ElephantInfo\* info, int numOfInfo)

{

int i;

int currentIndex=1;

//Longest 값을갖고 있는 인덱스를 구하는 과정입니다//

int max=table[1][numOfInfo];

int maxIndex=1;

for(i=0; i<numOfInfo; i++)

{

if(table[i+1][numOfInfo] > max)

{

max = table[i+1][numOfInfo];

maxIndex = (i+1);

}

}

printf("\nLongest Subsequence is %d\n\n",(max+1));

//

//조건을 검사하여 출력을 하는 부분입니다//

for(i=1; i<=numOfInfo; i++)

{

//printf("[i] is %d\n",i);

if(table[maxIndex][i]!=table[maxIndex][i-1])

{

printf("<<[%d]'s Elephant>>\n",info[currentIndex-1].index);

printf("Elephant's Weight is %d.\n",info[currentIndex-1].Weight);

printf("Elephant's IQ is %d.\n\n",info[currentIndex-1].IQ);

currentIndex=i;

}

if(table[maxIndex][i]==max)

{

printf("<<[%d]'s Elephant>>\n",info[currentIndex-1].index);

printf("Elephant's Weight is %d.\n",info[i-1].Weight);

printf("Elephant's IQ is %d.\n\n",info[i-1].IQ);

return;

}

}

//

}

void calculator(ElephantInfo\* info, int numOfInfo, int i, int j, int\* maxWeight,int\* minIQ)

{

if(i>=j)//이미 아이큐별로 솔팅해놓았기 때문에 의미없는 영역들(아이큐가줄어들일이 없는 영역들)

{

return;

}

if(info[j].Weight>\*maxWeight && info[j].IQ<\*minIQ)

{

table[i+1][j+1] = table[i+1][j]+1;

\*maxWeight = info[j].Weight;

\*minIQ = info[j].IQ;

}

else

{

table[i+1][j+1] = table[i+1][j];

}

}

void sortInfoByIQ(ElephantInfo\* info, int numOfInfo)

{

int i, j;

int Wtemp, Itemp, IdxTemp;

for(i=1; i<numOfInfo; i++)

{

for(j=0; j<(numOfInfo-i); j++)

{

if(info[j].IQ < info[j+1].IQ)

{

Itemp = info[j].IQ;

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

info[j+1].IQ = Itemp;

Wtemp = info[j].Weight;

info[j].Weight = info[j+1].Weight;

info[j+1].Weight = Wtemp;

IdxTemp = info[j].index;

info[j].index = info[j+1].index;

info[j+1].index = IdxTemp;

}

}

}

}

void insertInfo(ElephantInfo\* info, int numOfInfo)

{

int i;

//각 코끼리의 인덱스를 0으로 초기화 해주는 과정입니다//

for(i=0; i<numOfInfo; i++)

{

info[i].index=0;

}

//

//코끼리의 정보를 입력시키고 그 때 각 순서에 맞게 인덱스를 입력해주는 과정입니다//

for(i=0; i<numOfInfo; i++)

{

printf("Enter the %d's INFORMATION of ELEPHANT( :WEIGHT IQ)(1~10000)\n[Ex)7000 5000] : ", i+1);

scanf("%d %d",&info[i].Weight, &info[i].IQ);

info[i].index = (i+1);

if(info[i].Weight<1 || info[i].Weight>10000 || info[i].IQ<1 || info[i].IQ>10000)

{

printf("[ERROR] INFORMATION must between 1 to 10000!\n\n");

i--;

continue;

}

}

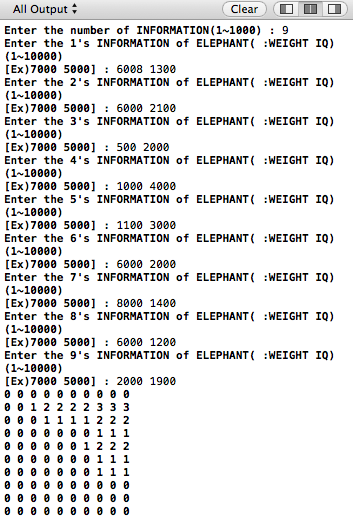
//

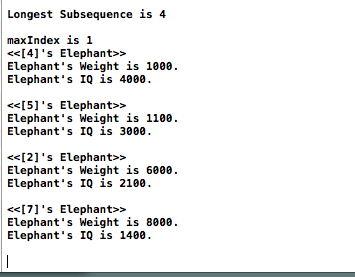
}

**//<테스트케이스는 다음장입니다>**

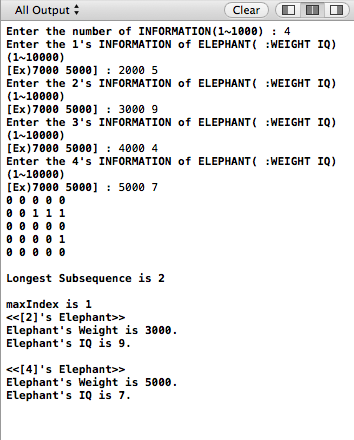
**// 저의 코드는 I Q솔팅입니다**

**1. 기본케이스 (강의자료 테스트 케이스)**

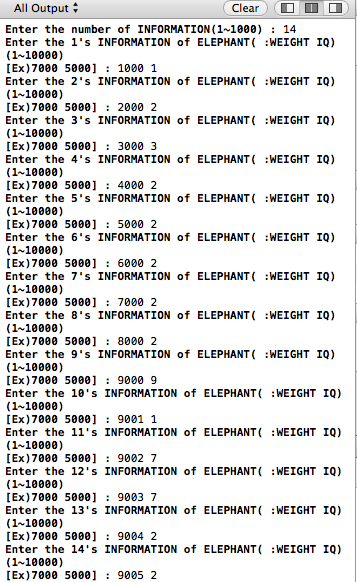
****

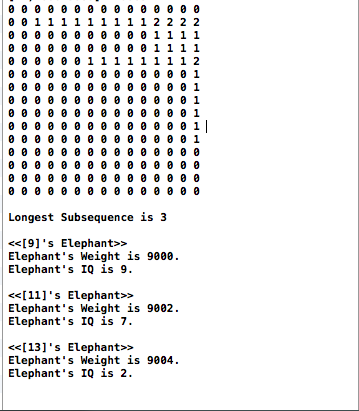
****

**2. 순서가 엇갈린 경우입니다**

****

**3. 많은 수를 넣은 경우입니다 (중복된 수도 넣어 보았습니다)**

**(다음장에 0이어서)**

****