**컴퓨터 알고리즘과 실습**

* 1회차 실습 -

2017.03.08

컴퓨터공학과

2013112149 박지용

문제 1-1.

<답> 1부터 n까지의 합을 구하고 거기서 1부터 n까지 중 어느 정수 하나가 제외된 상태의 합을 빼면 제외된 정수가 몇인지를 알 수 있다.

문제 1-2.

<답>

find\_except(input[], n)

for i ← 1 to n do

sum += i;

for i ← 0 to n-2 do

sum\_except += input[i];

except\_number ← sum – sum\_except;

return except\_number

문제 1-3.

<소스코드>

//2013112149 박지용

//문제 1번

#include <stdio.h>

int find\_except(int input[], int n) //input 배열과 정수의 개수(n)를 받아 제외된 수(x)를 찾음

{

int i, sum = 0, sum\_except = 0;

int x;

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

sum += i; //1~n까지의 합 구하기

for (i = 0; i < n - 1; i++) //1개가 제외되면 정수의 개수는 n-1개

sum\_except += input[i];

x = sum - sum\_except; //1~n까지의 합에서 숫자가 제외된 상태의 합을 빼서 제외된 숫자를 구함

return x;

}

int main(void)

{

int n = 0;

int input[] = { 5,6,2,4,9,10,8,7,1 }; //input배열의 크기가 n이 되어야함

printf("1~n까지의 정수 중 없는 하나를 찾고싶다. n을 입력하시오.\n");

scanf\_s("%d", &n);

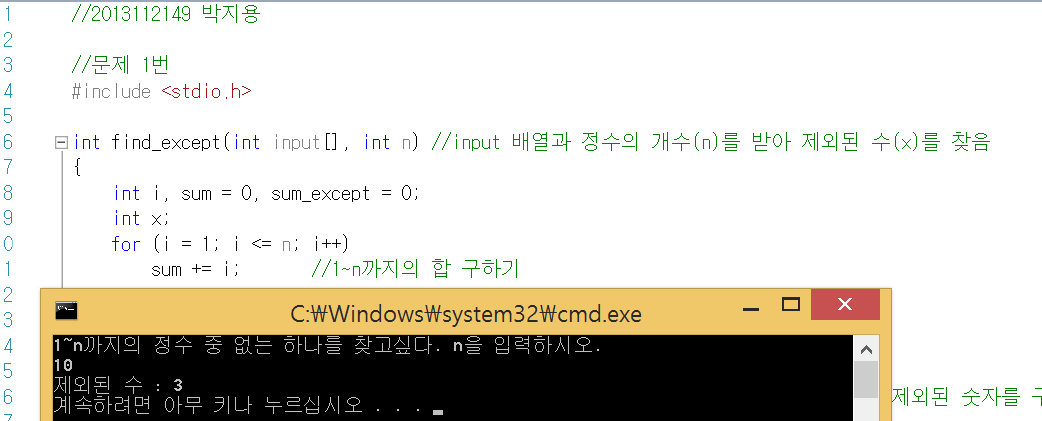
printf("제외된 수 : %d\n", find\_except(input, n));

return 0;

}

문제 1-4.

<답>



문제 2-1.  GreedyBetterChange 의 유사코드를 적어보아라. (5점)

<답>

GreedyBetterChange(M, c, d)

for i ← d to 1

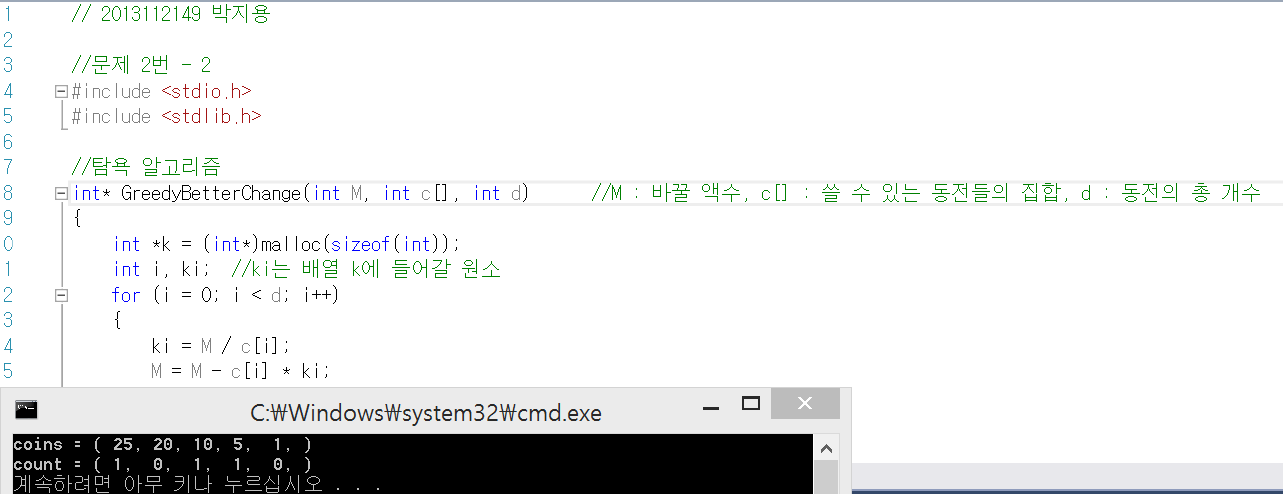
← M / ;

M ← M - \*;

return (;

문제 2-2.  GreedyBetterChange 를 프로그래밍 언어로 구현해 보고 M=40, c = (25,20,10,5,1) 에 대한 결과를 제출하시오. (10점)

<답>



k배열을 리턴값으로 받기 위해 포인터를 동적할당해서 리턴해줬습니다. 출력화면의 coins가 문제에선 배열 c이고 count가 배열 k입니다. 세부적인 내용은 주석으로 달아두었습니다.

문제 2-3.  BruteForceChange 의 유사코드를 적어보아라. (5점)

<답>

BruteForceChange(M, c, d)

smallest\_sum ← ∞;

for k ← {0, 0, ..., 0} to {M/, M/, ..., M/} do

howMuch ← ;

if howMuch = M

sum ← ;

if sum < smallest\_sum

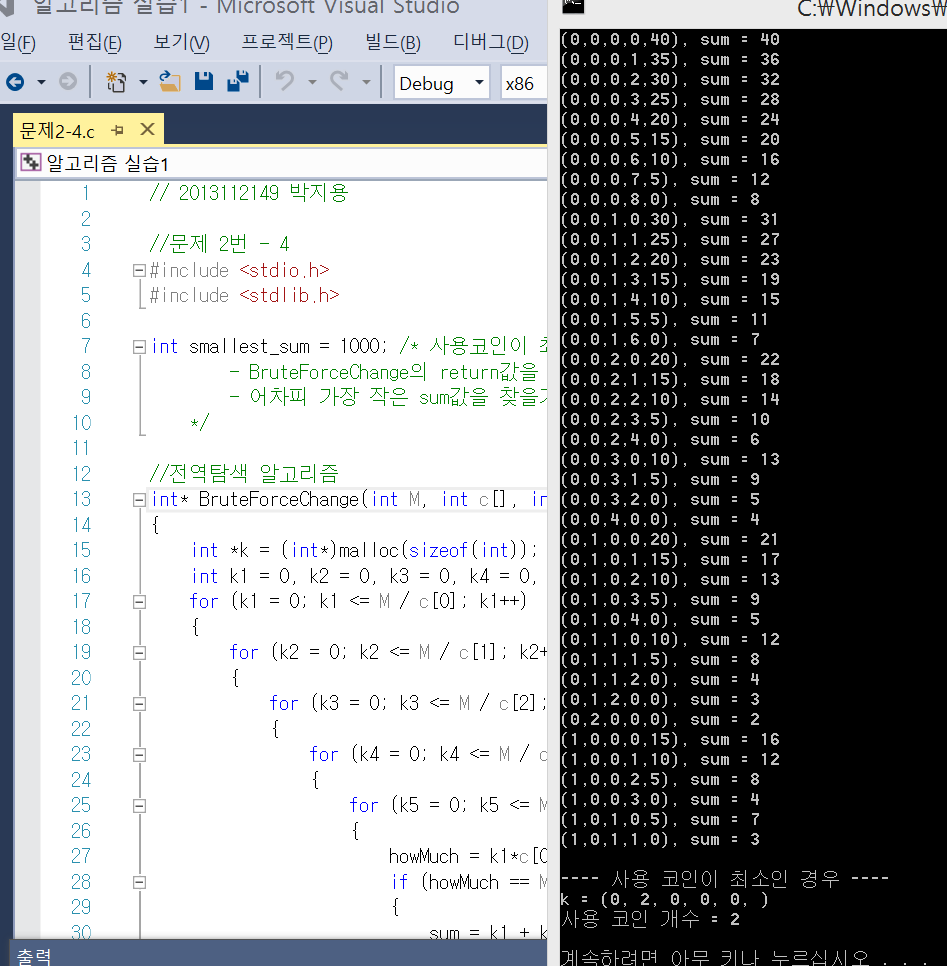
smallest\_sum ← sum;

best\_change = {;

return best\_change;

문제 2-4.  BruteForceChange 를 프로그래밍 언어로 구현해 보고 M=40, c = (25,20,10,5,1)에 대한 결과를 제출하시오. (10점)

<답>



5중 for문을 사용하여 k배열의 원소들을 차례차례 증가시키면서 모든 경우의 수를 찾는 식으로 구현하였습니다. k배열을 리턴값으로 받기 위해 포인터를 동적할당해서 리턴해줬습니다. 세부적인 내용은 코드에 주석으로 달아두었습니다.

문제 2-5.  어떤 M에 대해서도GreedyBetterChange가 correct한 결과를 주기위해 필요한 c6는 무엇인가? 가능한 값이 여러개일 경우 다 적으시오. (20점)

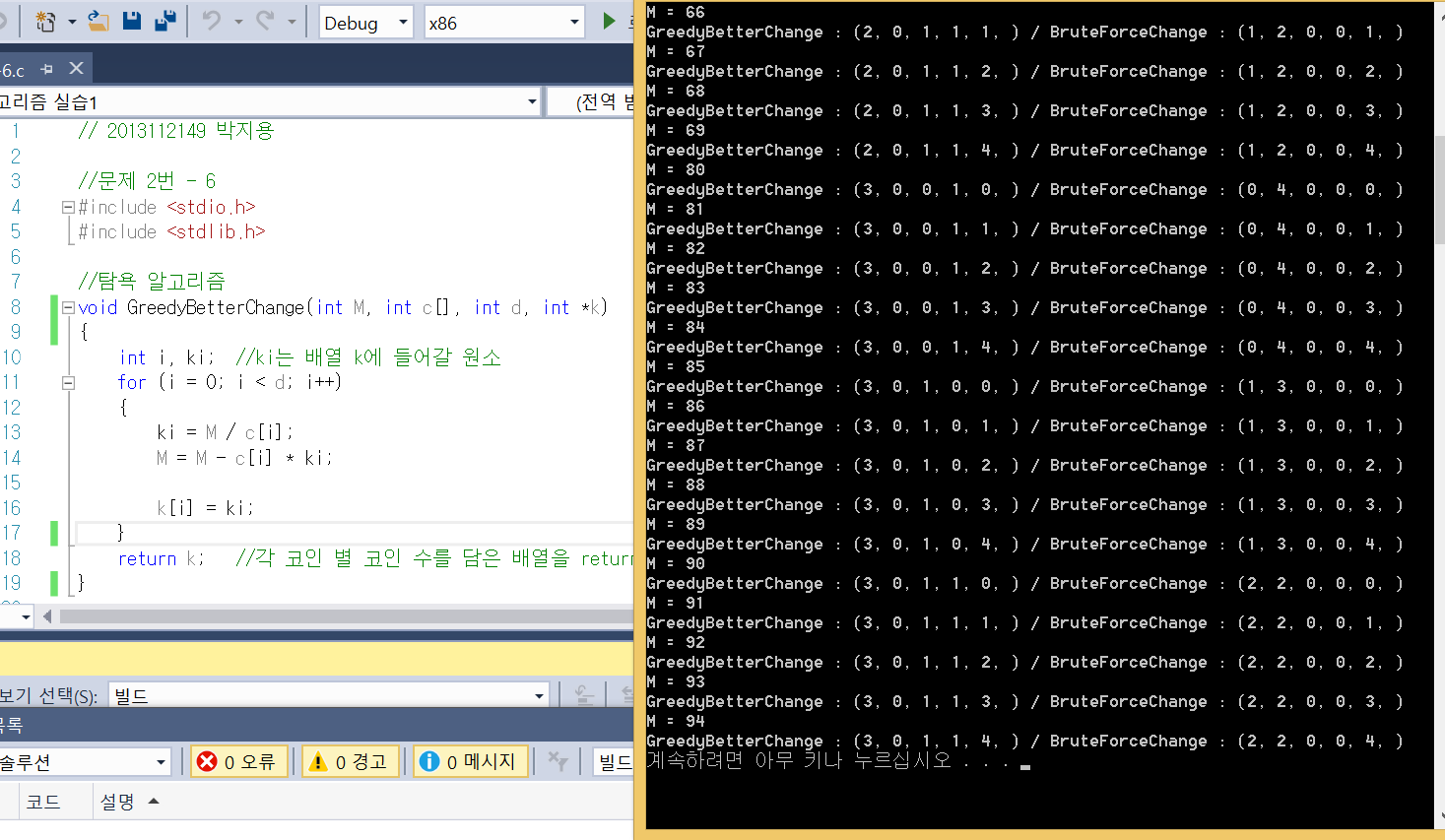
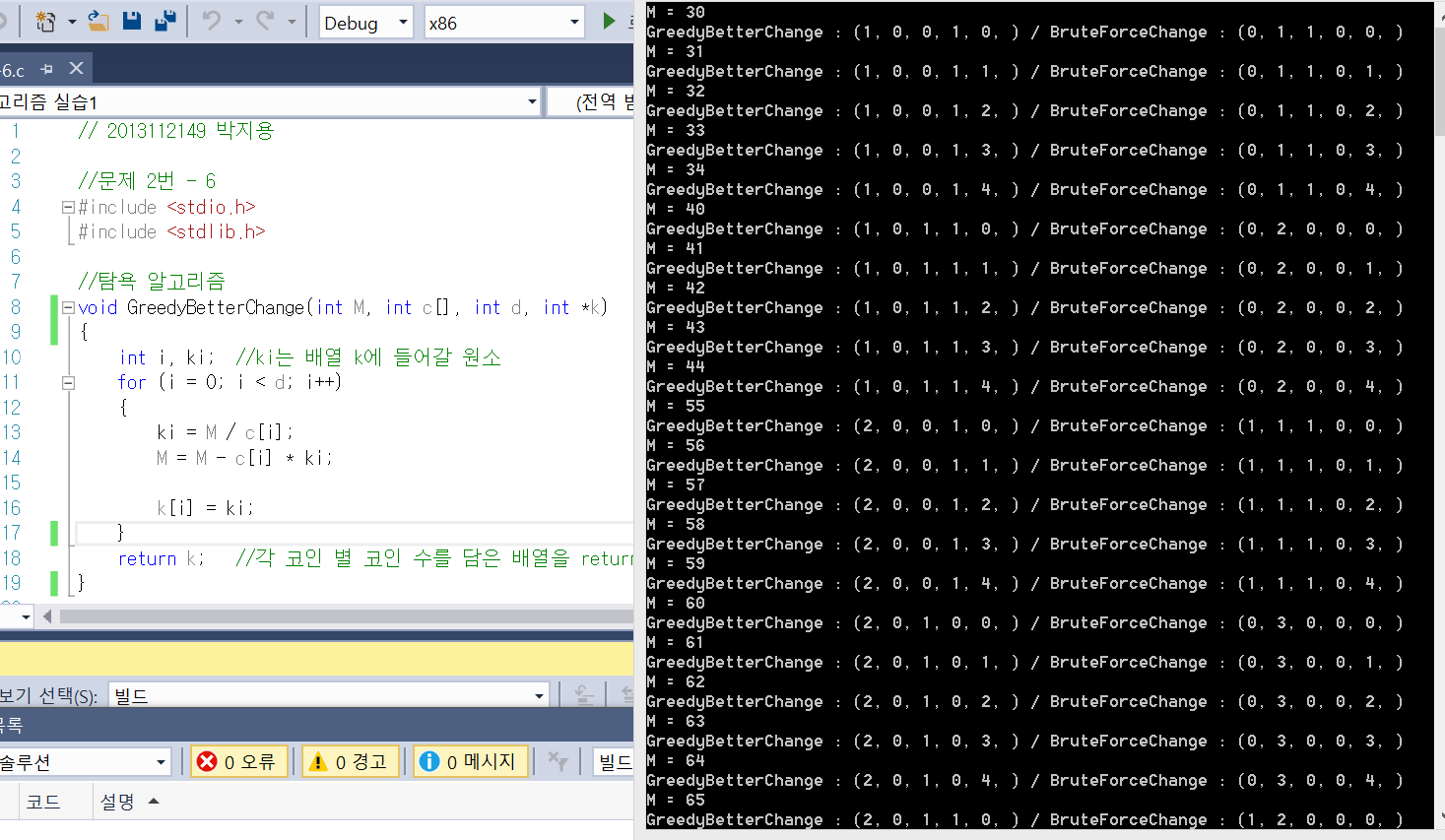
<답> = 15 혹은 40

<풀이> GreedyBetterChange 알고리즘이 항상 옳은 결과를 도출하기 위해서는 쓸 수 있는 최대의 동전이 우선적으로 사용되어야 한다. 즉, M보다 작은 동전 중 최대인 동전을 편의상 이라고 할 때, 은 최소 한번이상 쓰여야 한다. 문제의 크기를 줄이기 위해 M이보다 작은 수라고 하면 M값이 될 수 있는 수는 25 ~ 49가 된다. 이 중 두번의 연산 안에 끝낼 수 있는 수는 25, 26(25+1), 30(25+5), 35(25+10), 40(20\*2), 45(25+20) 의 4개인데, 여기서 25가 한번도 사용되지 않는 숫자는 40밖에 없다. 결국 40이라는 오류를 제거해야 하는데, 그러려면 코인 2개 안에 합이 40이 되어야한다. 이렇게 되는 경우는 25+15, 10+30, 5+35, 1+39, 40 의 경우가 있으므로 는 15, 30, 35, 39, 40 만 가능하고 M이보다 큰 경우로 확장시켰을 때 5개 중 안되는 수가 있을 수가 있으므로 각각에 대해 M이 40일 때의 GreedyBetterChange 알고리즘과 BruteForceChange 알고리즘을 비교하여 확인해보면 15와 40일 때만 차이가 없는 것을 확인할 수 있다.

문제 2-6.  c = (25, 20, 10, 5, 1) 를 이용하였을 경우와 5번에서 구한 를 포함한 6개의 동전을 이용하였을 경우 각각에 대해 M 을 1부터 100까지 대입해보면서 2번에서 구현한GreedyBetterChange 를 프로그램의 결과와 4번에서 구현한 BruteForceChange 의 결과를 비교하고 서로 다를 경우 이를 출력하는 프로그램을 작성해서 코드와 출력물 (화면 캡쳐)을 제출하시오. (20점)

<답>

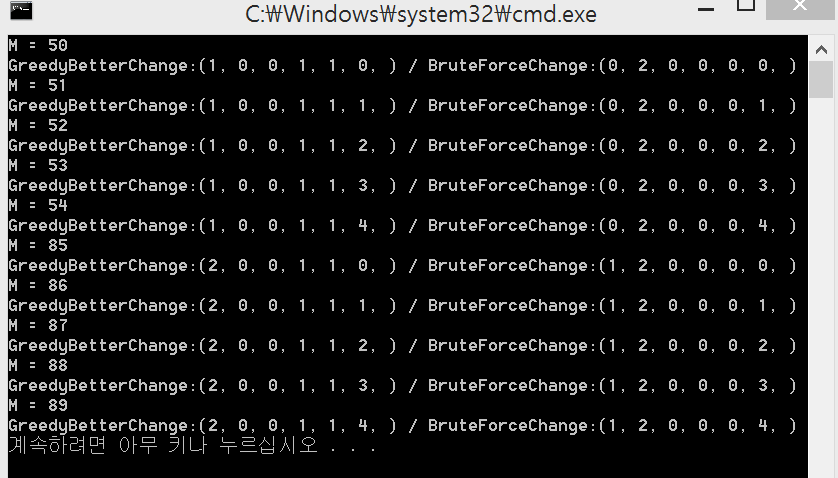
1. d = 5 일 때 – 조합이 다른 모든 경우를 출력하였음



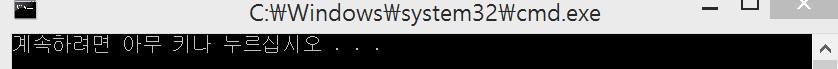
for문으로 M값을 증가시키면서 코인(k 배열)의 패턴이 다른 모든 경우에 대해서 출력하였습니다. 소스코드는 2-2번에서 작성한 GreedyBetterChange 함수와 2-4번에서 작성한 BruteForceChange 함수의 리턴값을 없게(void)하고 메인함수에서 동적할당한 포인터를 input으로 추가하는 식으로 변경하였습니다. 새로 array\_compare 함수를 추가해서 각각의 코인 수를 담은 배열(k1/k2)의 모든 원소가 같은지를 비교하였습니다.

1. d = 6 일 때 – 총 코인 사용 횟수의 최솟값이 다를 경우만 출력함

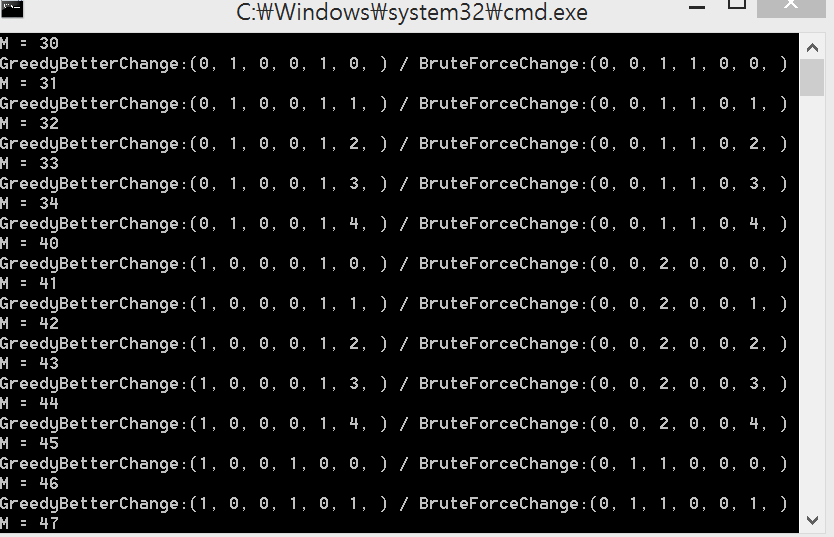
다음 출력화면은 GreedyBetterChange알고리즘이 항상 옳은 결과를 출력하도록 하는데적절치 못한 값(35를 사용)을 사용했을 때의 출력화면입니다.



다음 출력화면은 적절한 값을 사용했을 때의 출력화면입니다. 두 알고리즘이 다른 경우가 존재하지 않아 아무 출력결과도 나오지 않음을 확인할 수 있습니다.



여기서 array\_compare2함수가 아니라 array\_compare함수를 쓰면 다음 출력화면과 같이 코인이 사용된 횟수의 총합은 같지만 조합은 다른 경우를 모두 출력하게 됩니다. 출력화면은 일부만 캡쳐하였습니다. 이에 대한 설명은 아래에서 다시 하도록 하겠습니다.



<소스코드>

다음 소스코드는 d = 6일 때의 소스코드, 즉, 제가 찾아낸 이 추가되었을 때의 소스코드입니다.(d = 5일 때의 코드는 d값을 바꾸고 for문 하나만 줄이면 되므로 생략하였음) 자세한 설명은 주석으로 달아놓았지만 중요한 부분을 설명하자면 2-4번 코드에서 BruteForceChange알고리즘의 for문을 6중 for문으로 바꾸고 총 사용된 코인의 수가 같더라도 조합이 다르다면 모든 경우를 출력하는 array\_compare함수와 총 사용된 코인의 수가 다른 경우만을 출력하는 array\_compare2 함수를 만들어서 두 가지 방식으로 두 알고리즘을 비교할 수 있도록 하였습니다. 위에 있는 캡쳐화면 중 d = 5일 때의 캡쳐화면은 array\_compare함수를 쓴 화면이고 d = 6일 때는 출력이 비교적 짧은 array\_compare2함수를 쓴 화면을 첨부하였습니다. 현재 소스코드의 input은 값을 35로 두고 한 코드입니다. 이 값을 40이나 15로 고치면 두 알고리즘의 결과가 모두 같기 때문에 아무것도 출력되지 않습니다.

// 2013112149 박지용

//문제 2번 - 6

#include <stdio.h>

#include <stdlib.h>

//탐욕 알고리즘

void GreedyBetterChange(int M, int c[], int d, int \*k) //M : 바꿀 액수, c[] : 쓸 수 있는 동전들의 집합, d : 동전의 총 개수

{

int i, ki; //ki는 배열 k에 들어갈 원소

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

{

ki = M / c[i];

M = M - c[i] \* ki;

k[i] = ki;

}

return;

}

//전역탐색 알고리즘

void BruteForceChange(int M, int c[], int d, int \*k)

{

int smallest\_sum = 1000;

int k1 = 0, k2 = 0, k3 = 0, k4 = 0, k5 = 0, k6 = 0, sum = 0, howMuch = 0; //sum은 총 코인 사용 개수, howMuch는 각각의 코인수\*코인값을 다 더해서 나온 금액

for (k1 = 0; k1 <= M / c[0]; k1++) //5중 for문을 이용해 각 코인이 쓰인 횟수(k1, k2, k3, k4, k5, k6)를 늘려가면서 모든 경우의 수를 계산

{

for (k2 = 0; k2 <= M / c[1]; k2++)

{

for (k3 = 0; k3 <= M / c[2]; k3++)

{

for (k4 = 0; k4 <= M / c[3]; k4++)

{

for (k5 = 0; k5 <= M / c[4]; k5++)

{

for (k6 = 0; k6 <= M / c[5]; k6++)

{

howMuch = k1\*c[0] + k2\*c[1] + k3\*c[2] + k4\*c[3] + k5\*c[4] + k6\*c[5];

if (howMuch == M) // 금액이 주어진 금액과 같으면

{

sum = k1 + k2 + k3 + k4 + k5 + k6;

if (sum < smallest\_sum)

{

smallest\_sum = sum;

k[0] = k1; k[1] = k2; k[2] = k3; k[3] = k4; k[4] = k5; k[5] = k6;

//sum이 최소가 될 때의 코인이 쓰인 횟수(k1, ..., k6)를 배열에 저장

}

}

}

}

}

}

}

}

return;

}

//compare은 코인의 총 사용개수 뿐만 아니라 코인이 사용된 조합이 다른지를 비교하는 함수

int array\_compare(int a[], int b[], int d)

{

int i = 0;

while (i<d)

{

if (a[i] == b[i])

{

i++;

continue;

}

else

return 0;

}

return 1;

}

//compare2는 사용된 코인의 총 개수가 다른지를 비교하는 함수

int array\_compare2(int a[], int b[], int d)

{

int i, sum1 = 0, sum2 = 0;

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

{

sum1 += a[i];

sum2 += b[i];

}

if (sum1 == sum2)

return 1;

else return 0;

}

int main(void)

{

int M, d = 6, i;

int c[] = { 35, 25, 20, 10, 5, 1 };

int \*k1 = (int\*)malloc(sizeof(int));

int \*k2 = (int\*)malloc(sizeof(int));

for (M = 1; M < 100; M++)

{

GreedyBetterChange(M, c, d, k1);

BruteForceChange(M, c, d, k2);

if (!array\_compare2(k1, k2, d))

{

printf("M = %d\n", M);

printf("GreedyBetterChange : (");

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

{

printf("%d, ", k1[i]);

}

printf(") / BruteForceChange : (");

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

{

printf("%d, ", k2[i]);

}

printf(")\n");

}

}

return 0;

}