

1.HW_001

소스코드

```
#include <stdio.h>
#define _CRT_SECURE_NO_WARNINGS

int main()
{
    int min, max; //입력값
    int arr[10001] = { 0, }; //정수 갯수+끝 null
    int resultCount = 0; //실제 값

    printf("min :");
    scanf_s("%d", &min);
    printf("max :");
    scanf_s("%d", &max);

    for (int j = min; j <= max; j++)
    {
        arr[j] = j; //배열 "min"번 부터 "max"번까지를 각각 숫자로 채움
    }
    for (int k = min; k <= max; k++)
    {
        int index = 1; //제공인지 검사 변수
        for (int num = 2; num * num <= arr[k]; num++)
            //num이 1씩 커지면서 배열이 num의 제공인지 검사,
            //배열의 숫자보다 더 큰수로 나누면 무조건 나머지는 0이 아니기 때문에
            num^2이 배열보다 작거나 같을때까지
        {
            if (arr[k] % (num * num) == 0) //제공수로 나누었을때 나머지가 0이 되는
            경우
            {
                if (arr[k] / (num * num) == 1) //제공수를 소인수로 가지면서 제공이
                아닌경우를 제외
                //ex) 12=2x2x3
                {
                    index = 0; //제공일때를 갯수세기에서 제외
                    break;
                }
            }
        }
        if (index == 1) //제공이 아니라면
        {
            if (k != 1) //1은 제공x
```

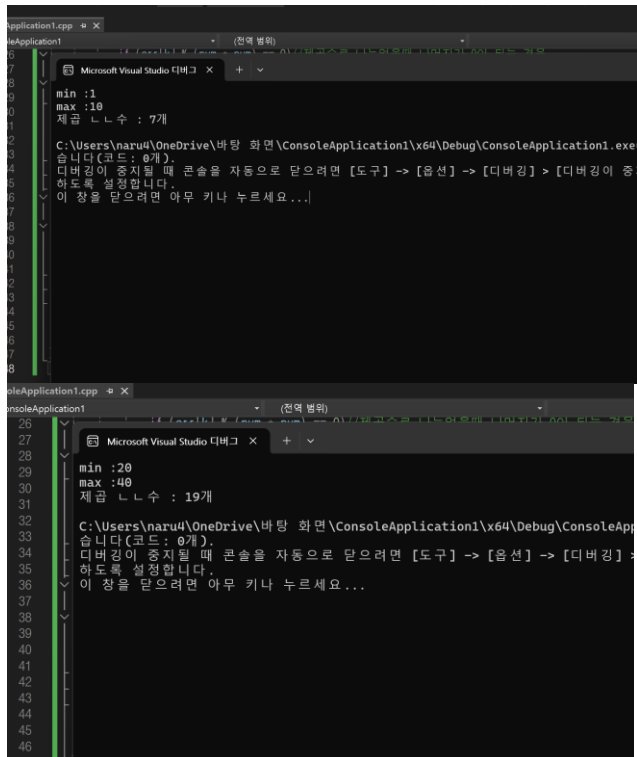
```

        {
            resultCount++; //갯수+1
        }
    }
}

printf("제공 나눔수 : %d개\n", resultCount);
return 0;
}

```

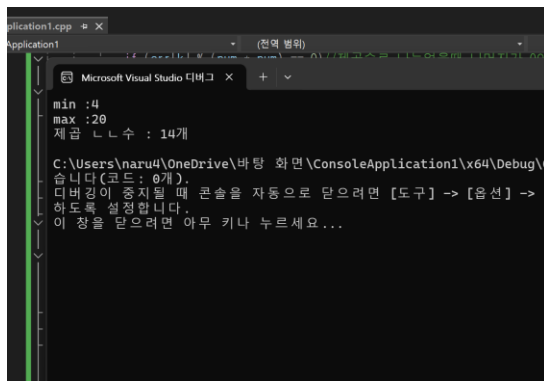
실행결과



```

Application1.cpp  X
Application1 (전역 범위)
Microsoft Visual Studio 디버그 X +
min :1
max :10
제공 나눔수 : 7개
C:\Users\naru4\OneDrive\바탕 화면\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe(
습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [디버깅] > [디버깅이 중지
하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...]

```



```

Application1.cpp  X
Application1 (전역 범위)
Microsoft Visual Studio 디버그 X +
min :4
max :20
제공 나눔수 : 14개
C:\Users\naru4\OneDrive\바탕 화면\ConsoleApplication1\x64\Debug\C
습니다(코드: 0개).
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구] -> [옵션] -> [
하도록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요...]

```

HW_002

소스 코드

```
#include<stdio.h>
#include<string.h>
#define _CRT_SECURE_NO_WARNINGS

int main() {
    int set[21] = { 0, }; //집합 배열
    int num; //숫자 입력 변수
    char op[10] = { 0, }; //연산 변수
    int index = 0; //배열의 주소 변수
    printf("연산을 선택하세요. (1 <= x <= 20)\n");
    printf("add X\nremove X\ncheck X\ntoggle X\nall O\nempty O\n");

    while (1) //무한 반복
    {
        printf("\ninput : ");
        scanf_s("%s", op, 30);
        scanf_s("%d", &num);

        if (strcmp(op, "add") == 0) //add의 경우
        {
            int check1 = 0;
            for (int i = 0; i < index; i++) //중복 검사
            {
                if (set[i] == num) //배열중 중복되는 숫자가 있다면 연산을 무시
                {
                    check1 = 1;
                    break;
                }
            }
            if (check1 == 0)
            {
                set[index] = num; //자리가 비어있는 가장 앞번 주소부터 채움
                index++; //배열을 채웠기 때문에 주소+1
            }
        }

        else if (strcmp(op, "remove") == 0)
        {
            for (int i = 0; i < index; i++) //숫자 검사
            {
```

```

        if (set[i] == num)//삭제하려는 숫자의 주소==i
        {
            for (int k = i; k <= index - 1; k++)//배열의 끝주소 전 까지 반복
            {
                set[k] = set[k + 1];//배열의 숫자를 한칸씩 앞으로 당기기
            }
            index--;//숫자 하나가 사라졌기때문에 주소 -1
            break;
        }
        else//삭제하려는 숫자가 없을때 연산 무시
            break;
    }
}

```

```

else if (strcmp(op, "check") == 0)
{
    int numCheck = 0;//숫자가 검사되지 않았다면 0출력
    for (int i = 0; i < index; i++)//배열 검사
    {
        if (set[i] == num)
        {
            numCheck = 1;//숫자가 검사되었다면 1출력
            break;
        }
        else
            break;
    }
    printf("%d\n", numCheck);
}

```

```

else if (strcmp(op, "toggle") == 0)
{
    int check2 = 0;
    for (int i = 0; i < index; i++)
    {
        if (set[i] == num)
        {
            for (int k = i; k < index - 1; k++)//숫자가 있다면
            {
                set[k] = set[k + 1];
            }
            index--;
            check2 = 1;
        }
    }
}

```

'remove알고리즘'

```

        break;
    }
}
if (check2 == 0)
{
    set[index] = num; //숫자가 없다면 'add알고리즘'
    index++;
}
}

else if (strcmp(op, "all") == 0)
{
    for (int i = 0; i < 20; i++)
    {
        set[i] = i + 1; //20까지 배열의 각 자리에 숫자 채우기
    }
    index = 20; //배열 주소는 20까지 밀기
}

else if (strcmp(op, "empty") == 0)
{
    for (int i = 0; i < 20; i++) //배열내 숫자 전부 초기화
    {
        set[i] = 0;
    }
    index = 0; //다 삭제 되었으므로 배열 주소는 0
}

else
{
    continue; //연산자가 잘못입력된 경우 그냥 무시
}

printf("집합 : {"); //출력의 앞 "집합 : {"구현
for (int i = 0; i < index; i++)
{
    printf("%d", set[i]);
    if (i < index - 1)
    {
        printf(", ");
    }
}
printf(" }\\n"); //출력의 뒤 "}"구현
}

```

}

실행결과

```
C:\Users\Wnaru4WOneDriveWl X + v
연산을 선택하세요. (1 <= x <= 20)
add X
remove X
check X
toggle X
all 0
empty 0

input : add 5
집합 : {5 }

input : remove 5
집합 : { }

input : all 0
집합 : {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

input : remove 3
집합 : {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

input : remove 5
집합 : {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

input : check 3
0
집합 : {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 }

input : empty 0
집합 : { }
```

```
input : add 3
집합 : {3 }

input : add 4
집합 : {3, 4 }

input : add 5
집합 : {3, 4, 5 }

input : toggle 6
집합 : {3, 4, 5, 6 }

input : toggle 2
집합 : {3, 4, 5, 6, 2 }

input : toggle 3
집합 : {4, 5, 6, 2 }

input :
```

Read me: 실행화면상에서 보이듯이 대부분 작동하지만 왜 인지 remove가 작동 안 할때도 있네요. 예외상황도 고려한 코드를 만들고 싶었지만 조건을 넣는 순간 코드가 작동을 안해 버려서 배열이 20개를 넘어가는 상황이나 1또는 20을 넘어가는 숫자를 입력한 상황 두개를 고려하지 못했습니다. 과제 2번을 하면서 앞에서 질문 받아주신 선배님의 조언이 큰 도움이 되어 감사하다는 말씀드리고 싶습니다.