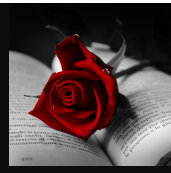
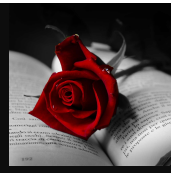


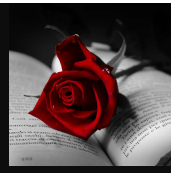
20190905
컴퓨터프로그래밍 및 실습 1주차

박 건 호
devgunho.github.io



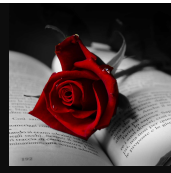
C++은 크게 4가지로 나누어진다.

- C Language
- OOP (Object Oriented Programming)
- STL (Standard Template Library)
- Metaprogramming



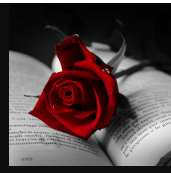
강의 순서

- 비주얼 스튜디오 활용법 익히기
- C 복습 (자료형)
- C 복습 (소수찾기문제)
- C++이 어떻게 변하고 있는지
- C++ 기본 출력
- C++ 기본 입력
- C++ 문자와 문자열 입력방법



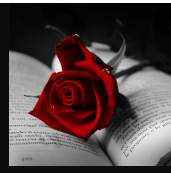
비주얼 스튜디오 활용하기

- 언어 관리
- 프로젝트 생성
- 솔루션 파일 / 사용자 파일
- 소스코드 개별 빌드 및 관리하기
- Import / Export



C 복습 (자료형)

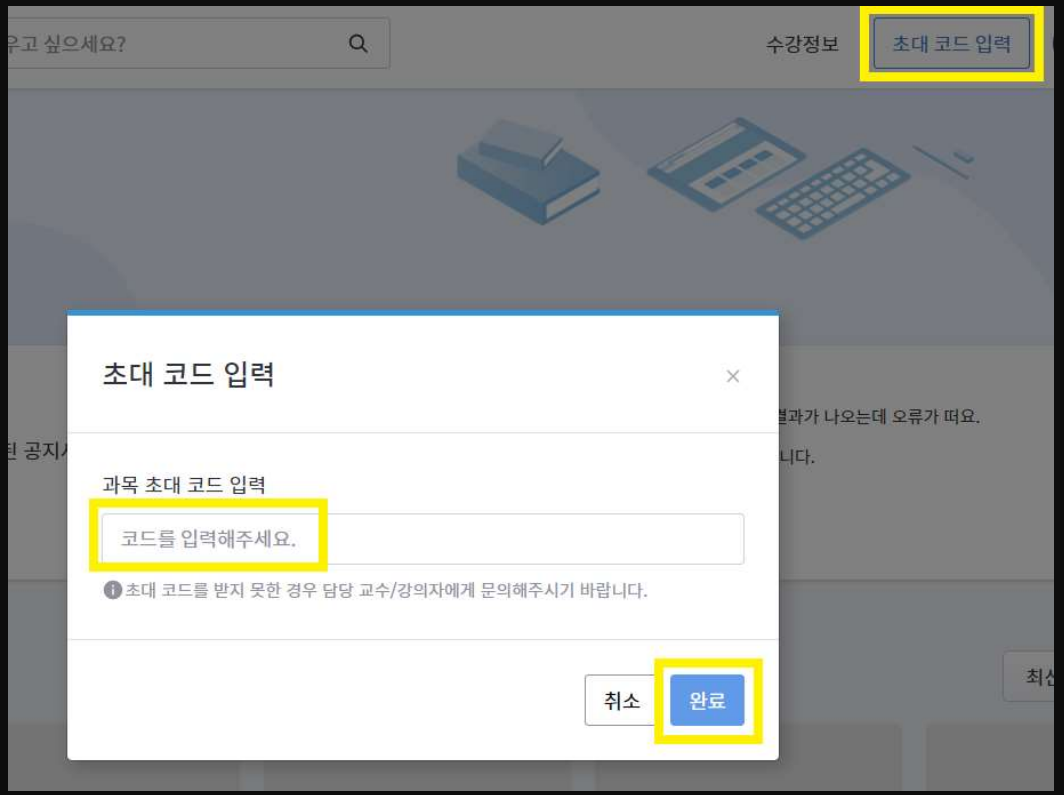
```c
#include <stdio.h>

int main()
{
    printf("정수 형식의 크기 확인\n");
    printf("char : %d \n", sizeof(char));
    printf("unsigned char : %d \n", sizeof(unsigned char));
    printf("short : %d \n", sizeof(short));
    printf("unsigned : %d \n", sizeof(unsigned short));
    printf("int : %d \n", sizeof(int));
    printf("unsigned int : %d \n", sizeof(unsigned int));
    printf("long : %d \n", sizeof(long));
    printf("unsigned long : %d \n", sizeof(unsigned long));
    printf("long long : %d \n", sizeof(long long));
    printf("double long : %d \n", sizeof(double long));
    return 0;
}
```

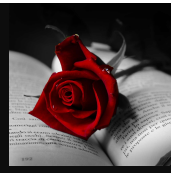


C 복습 (자료형)

형식명	바이트 수	표현 범위
char	1	-128~127
unsigned char	1	0~255
short	2	-32768~32767
unsigned short	2	0~65536
int	4	-2,147,483,648~2,147,483,647
unsigned int	4	0~4,294,967,295
long	4	-2,147,483,648~2,147,483,647
unsigned long	4	0~4,294,967,295
long long	8	-9223372036854775808 ~9223372036854775807
unsigned long long		0 ~ 18446744073709551615

과제 제출방법 – 구름 IDE / hufs.goorm.io





과제1 (10점) 정수를 입력 받고 다양하게 출력하기

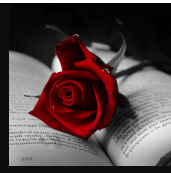
printf(“%d”,output);

- ☞ %d는 정수를 출력할 때 사용
- ☞ %c는 문자를 출력할 때 사용 …

단순히 출력형식들을 시험기간 때 잠간 외워 영원히 잊어버리지 말고,

직접 코드로 작성해 보고 오래도록 기억하는 것을 목표로 한다.

C 복습 (반복문)



초기식 조건식 변화식

```
for ( int i = 0 ; i < 100 ; i++ )
{
    printf("Hello, world!\n");
}
```

반복할 코드

초기식

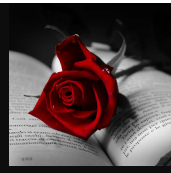
```
int i = 0;
```

조건식

```
while ( i < 100 )
{
    printf("Hello, world!\n");
    i++;
}
```

변화식

반복할 코드



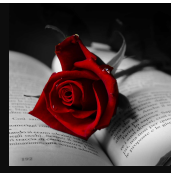
C 복습 (소수 찾기)

자신보다 작은 두 개의 자연수를 곱하여 만들 수 없는 1보다 큰 자연수이다.

예를 들어, 5는 1x5 또는 5x1로 수를 곱한 결과를 적는 유일한 방법이 그 수 자신을 포함하기 때문에 5는 소수이다.

그러나 6은 자신보다 작은 두 숫자(2×3)의 곱이므로 소수가 아닌데, 이렇듯 1보다 큰 자연수 중 소수가 아닌 것은 **합성수**라고 한다.

1과 그 수 자신 이외의 자연수로는 나눌 수 없는 자연수로 정의하기도 한다.



C 복습 (소수 찾기)

원하는 출력값 :

1 : 2

2 : 3

3 : 5

4 : 7

5 : 11

6 : 13

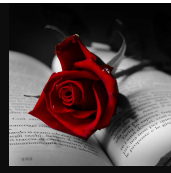
7 : 17

8 : 19

... (계속 출력...)

C 복습 (소수 찾기)

```c
#include <stdio.h>

int main(void) {
    int i, j;
    int cnt = 0;

    for (i = 1; ; i++) {
        for (j = 2; j <= i; j++)
        {
            if (i % j == 0) break; // 반복중에 i를 j로 나눈값이 0이 된다면 탈출
        }

        // 반복문을 탈출했는데 i와 j가 같다는 것은 자기자신으로 나눴을때만 0이 나온다는 것
        // 즉, 1과 나 자신 숫자 외에는 나눌수있는 수가 없다는 것
        if (i == j)
        {
            cnt++;
            printf("%d : %d\n", cnt, j);
        }
    }

    return 0;
}
```



과제2 (13점) 5만 번째 소수 찾기

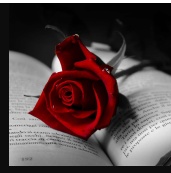
아래 힌트를 생각해보자.

"어떤 수 X가 소수임을 증명하는데 있어서,
1부터 X까지 모두 나눠볼 필요가 있을까?"

에라토스테네스의 접근

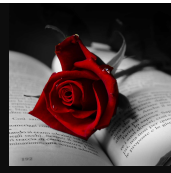
주어진 자연수 N이 소수이기 위한 필요 충분조건은 N이 N의 제곱근보다 크지 않은 어떤 소수로도 나눠지지 않는다.

수가 수를 나누면 몫이 발생하게 되는데 몫과 나누는 수, 둘 중 하나는 반드시 sqrt(N) 이하이기 때문이다.



C++

1979 | C with Classes : Simula라는 객체 지향적 언어의 컨셉을 접목
1983 | C++ : 가상 함수, 연산자 오버로딩, 참조, const, new/delete 연산자 등등 추가
1989 | C++ 2.0 : 다중 상속, 추상 클래스, static 멤버 함수, const 멤버함수 등등 추가
1998 | C++98 : 프로그래밍 언어학자 및 컴파일러 제조사들을 모아 표준화 작업 시작
2003 | C++03 : 표준화 문서상 불명확했던 것을 교정한 버전
2011 | C++11 : 람다 표현식, 스마트 포인터, 정규표현식, 멀티쓰레드 등등 추가
2014 | C++14 : C++11에 추가된 요소들을 다듬고 확장하는데 치중한 마이너 표준안
2017 | C++17 : 파일 시스템, 알고리즘 병렬 처리, 두개 이상의 값을 반환하는 기능 등등
2020 | C++20 : 비동기 프로그래밍 지원, 컴파일 속도 향상을 위한 소스 파일 모듈화, 등등



C++

1979 | C with Classes : Simula라는 **객체 지향적 언어**의 컨셉을 접목
1983 | C++ : **가상 함수**, **연산자 오버로딩**, **참조**, **const**, **new/delete** 연산자 등등 추가
1989 | C++ 2.0 : **다중 상속**, **추상 클래스**, **static 멤버 함수**, **const 멤버함수** 등등 추가
1998 | C++98 : 프로그래밍 언어학자 및 컴파일러 제조사들을 모아 표준화 작업 시작
2003 | C++03 : 표준화 문서상 불명확했던 것을 교정한 버전
2011 | C++11 : **람다 표현식**, **스마트 포인터**, **정규표현식**, **멀티쓰레드** 등등 추가
2014 | C++14 : C++11에 추가된 요소들을 다듬고 확장하는데 치중한 마이너 표준안
2017 | C++17 : 파일 시스템, 알고리즘 병렬 처리, 두개 이상의 값을 반환하는 기능 등등
2020 | C++20 : **비동기 프로그래밍** 지원, **컴파일** 속도 향상을 위한 소스 파일 **모듈화**, 등등

C++ “Hello World!”

test.cpp

```
// 첫번째 예제 프로그램
#include <iostream>
using namespace std;

int main(void)
{
        cout << "Hello World!" << endl;
        return 0;
}
```

주석

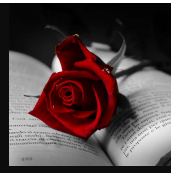
헤더파일

이름공간 설정

화면에 문자열 출력

실행결과

```
Hello World!
```

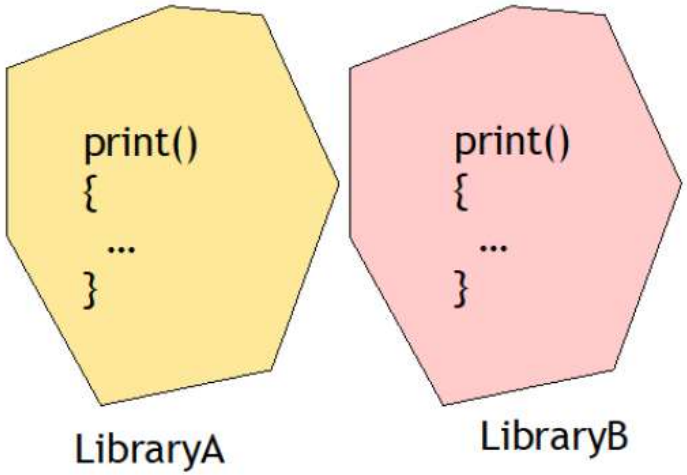


C++ “Hello World!”

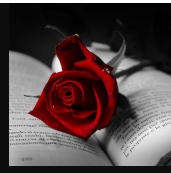
using namespace std;

INFINITY BOOKS 인피니티북스

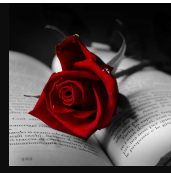
- 변수 이름이나 함수 이름과 같은 수많은 이름(식별자)들은 이름 공간(name space)이라고 하는 영역으로 분리되어 저장



C++ “Hello World!”



```cpp
#include <iostream>

using namespace std;    // using은 이름 공간을 지정하는 지시어이다.
    // 추후 클래스의 private 키워드와 함께 현업에서 쓰이는 규모가 큰 프로그램을 예시로하여
    // 꼭 이름 공간을 구분지어주어야 하는 예시에 대해서 강의하도록 하겠습니다.

int main()
{
    cout << "Hello World!" << endl;
    // cout은 I/O-stream library가 제공하는 객체로서 출력을 담당
    return 0;
}
```

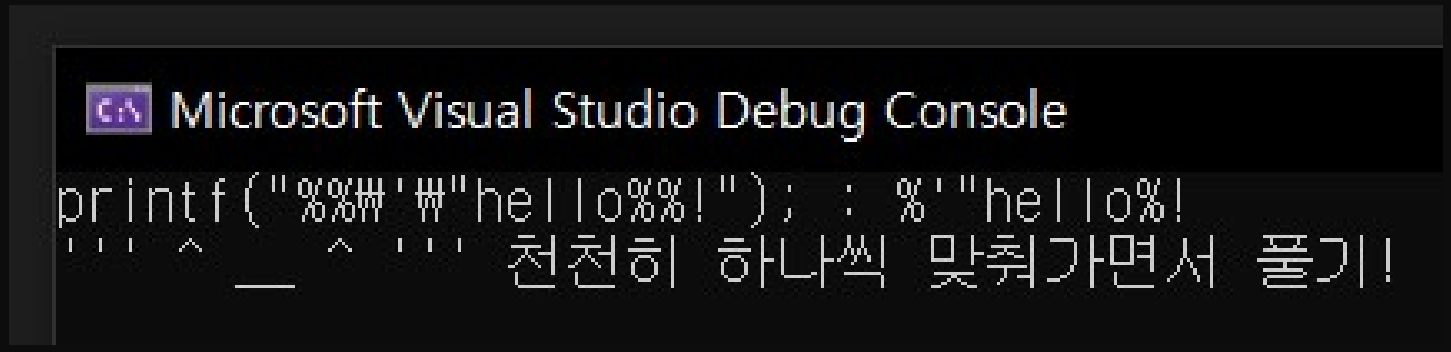


과제3 (10점) 출력 마스터하기

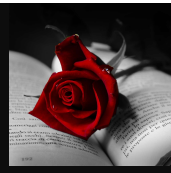
출력:

printf("%%₩"₩"hello%%!"); : %'"hello%!

'" ^ __ ^ "' 천천히 하나씩 맞춰가면서 풀기!



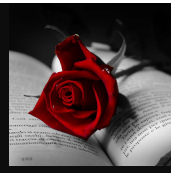
C++ “Hello World!”



출력 객체 cout

```
cout << “Hello World!” << endl;
```

- cout은 IO-stream library가 제공하는 객체로서 출력을 담당합니다.
- 큰따옴표 안의 문자열을 화면에 출력합니다.
- endl은 문장의 끝을 나타내는 기호(\n로 정의되어 있다).



C++ “Hello World!”

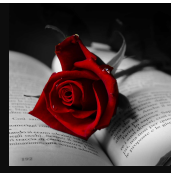
정말 <iostream>에서 cout이라는 객체를 제공하고 있을까?

ostream std::cout - C:₩Program Files (x86)₩Microsoft Visual Studio₩2019₩Community₩VC₩Tools₩MSVC₩14.22.27905₩include₩iostream(29)

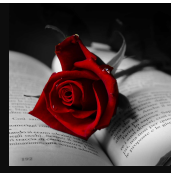
```
__PURE_APPDOMAIN_GLOBAL extern ostream cout, *_Ptr_cout;
```

```
__PURE_APPDOMAIN_GLOBAL extern _CRTDATA2_IMPORT ostream cout, *_Ptr_cout;
```

출력 객체 : cout

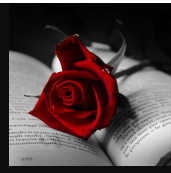


- cout<<출력대상<<출력대상<<출력대상<<.....;
- '<<'는 put to라고 읽는다.
- ostream 클래스로 만들어진 객체이다.
- cout을 이용하면 기술한 순서대로 출력하게 된다.
- cout에 의해 자동으로 변수나 상수의 자료형 검사가 이루어지고 출력 된다.



입력 객체 : cin

- cin>>변수>>변수>>변수>>.....;
- '>>'는 get from이라고 읽는다.
- istream 클래스로 만들어진 객체이다.
- cin을 이용하면 기술한 순서대로 입력 받게 된다.
- 키보드에서 입력된 데이터는 cin에 의해 자동으로 변수의 자료형 검사가 이루어지고 저장된다.
- 하나 이상의 변수를 나열해서 자료를 입력 받는 경우, 각 데이터는 공백문자와 enter로 구분한다.



입력 객체 : cin

cin >>i >>s >>w;

입력 방법 :

① 11 22 33↵

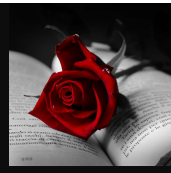
② 11↵

22↵

33↵

입력 객체 : cin

```
1   #include <iostream>
2
3   using namespace std;
4
5   int main()
6   {
7       int a, b, c;
8       cin >> a >> b >> c;
9       cout << a << endl;
10      cout << b << endl;
11      cout << c << endl;
12      return 0;
13  }
14
```

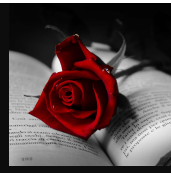


문자와 문자열 입력방법

(1) cin 이용
- 문자, 문자열, 정수, 실수를 입력 받는다.
- spacebar, enter, tab key는 받아들이지 않는다.

(2) cin.get(인수1, 인수2);
- 문자열을 입력받는다.
- 인수1 : 배열의 주소
  인수2 : 입력받고자 하는 문자열의 최대 길이 (배열의 크기보다 작거나 같아야 한다.)
  입력받을 수 있는 문자열의 최대 크기는 (인수2 - 1)로 입력받는다.
  제일 끝에는 NULL문자용이다.
- 키보드 버퍼를 사용하기 때문에, enter를 치면 문자열을 입력을 받는다.
  그러나, enter값은 입력 받지 않고 버퍼에 남겨놓는다. 그래서 다음 입력함수에 영향을 준다.

(3) fflush(stdin)
- 키보드 버퍼를 비운다.



문자와 문자열 입력방법

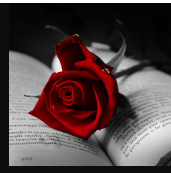
cin.get() 함수는 문자열 data만 name에 저장시키고 enter(₩n) 값은 그대로 키보드 버퍼에 남겨둔다.

그래서 다음에 있는 cin.get()는 enter 값을 입력받는다.

fflush(stdin) 함수는 키보드 버퍼의 문자를 제거하는 역할을 한다.

그래서, 다음에 입력받는 cin.get()함수에 영향이 미치지 않게 한다.

~~BOF 공격~~



문자와 문자열 입력방법

(1) cin 이용

- 문자, 문자열, 정수, 실수를 입력 받는다.
- spacebar, enter, tab key는 받아들이지 않는다.

(2) cin.getline(인수1, 인수2); 을 사용하자!

문자와 문자열 입력방법

```cpp
#include <iostream>

using namespace std;

void main()
{
    char name[10];
    char major[10];

    cout << "What's your name? : ";
    cin.getline(name, 10);
    fflush(stdin);

    cout << "major? ";
    cin.getline(major, 10);
    cout << "\n\nName : " << name << " Major : " << major << endl;
}
```