

Lab 2

Introduction of Programming Environment

Basic of C++

- scan and print
- `cout << 1 << 'a' << "String" << 두0|;`
- `cin >> val1 >> val2;`

Function Overloading

```
#include <iostream>

void function(void)
{
    std::cout << "function(void) call" << std::endl;
}

void function(int a, int b)
{
    std::cout << "function("<<a<<","<<b<<") call"<<std::endl;
}

int main(void)
{
    function();
    function(12, 13);
    return 0;
}
```

! Same name functions : Error occur on C

Default Parameter

```
#include <iostream>

using std::cout;
using std::endl;

int func(int a = 0)
{
    return a+1;
}

int main(void)
{
    cout << func(11) << endl;
    cout << func() << endl;
    return 0;
}
```

func() parameter: int type data & default parameter = 0

In-line function

```
#include <iostream>

inline int SQUARE(int x)
{
    return x*x;
}

int main(void)
{
    std::cout << SQUARE(5) << std::endl;
    return 0;
}
```

Function inline : Swap function call statement to function's body

C++ Basics & Data Types

- C++ : 소프트웨어 개발 플랫폼에 객체 지향 개념 제공
- 객체지향 : 자료와 이들 자료를 어떻게 다룰 것인지 따로 생각하지 않고 하나의 사물로 생각

C++ Basics & Data Types

- Class라는 새로운 형 사용
 - 변수가 사용하는 메모리 크기
 - 변수가 가질 수 있는 정보
 - 변수에게 가능한 조작
- Class의 정의
 - 변수들과 연관된 함수들을 결합시킨 새로운 형
 - 클래스를 선언함으로써 새로운 형 생성
 - 구조체를 선언하는 것도 새로운 형을 생성하는 것이지만 함수까지 결합 한다는 것이 구조체와 클래스의 차이
 - 구조체에 함수의 기능을 추가시킨 자료구조

C++ Basics & Data Types

- 용어
 - 멤버 변수 : 클래스 내의 변수
 - 멤버 함수, 메소드 : 클래스 내의 함수, 객체가 무엇을 하는지를 결정
 - 객체 : 클래스에 의해 만들어진 변수

C++ Basics & Data Types

- 클래스 선언
- class 클래스명

```
{  
    Member variable;  
    Member fuction;  
};
```

```
Ex)  
class Fishbread  
{  
    string content;  
    void Wrapped();  
};
```

클래스 선언 시 메모리 할당이 되는 것이 아니라
객체를 생성할 때 메모리 할당이 일어남.

C++ Basics & Data Types

- 객체 정의
- 클래스를 실체화

Ex)

```
Fishbread fish1("팥");  
Fishbread *fish2=new Fishbread("슈크림");
```

- 객체 멤버에 접근하기

Ex)

```
fish1.content="딸기";  
fish2.wrapped();
```

Ex)

```
#include <iostream>  
#define WELL_DONE 3;  
class Fishbread  
{  
public:  
    string content;  
    int roasting;  
};  
int main()  
{  
    Fishbread fish1;  
    fish1.content="슈크림";  
    fish1.roasting=WELL_DONE;  
    std::cout<<"붕어빵은 "<<fish1.content<<"로 만들어 졌으  
며"<<fish1.roasting<<"정도로  
구워짐\n";  
    return 0;  
}
```

C++ Basics & Data Types

- 접근제어 지시자
 - public : 어디서든 접근 허용
 - protected : 상속관계에 놓여 있을 때, 유도 클래스에서의 접근 허용
 - private : 클래스 내(클래스 내에 정의된 함수)에서만 접근 허용
 - 명시되지 않은 경우에 대해서는 default로 public
- 클래스에서 private은 외부로 객체의 data를 마음대로 접근할 수 없도록 하기 위해(캡슐화) 사용하며, 캡슐화된 데이터에 접근하기 위해서는 public으로 선언된 메소드를 선언

C++ Basics & Data Types

Ex)

```
class Fishbread
{
    private:
        int cost;
        int seller;
        string content;
        int roasting;
    public:
        int GetCost();
        void SetCost(int cost);
};

...
fish1.SetCost(500);
cout<<"가격은?"<<fish1.GetCost()<<endl;
fish1.SetCost(700);
cout<<"가격은?"<<fish1.GetCost()<<endl;
```

C++ Basics & Data Types

- 클래스 메소드의 구현

리턴형 클래스명::함수명(매개변수1,매개변수2,...)

```
{  
    ...  
}
```



```
class Fishbread  
{  
    private:  
        int cost;  
        int seller;  
        string content;  
        int roasting;  
    public:  
        int GetCost();  
        void SetCost(int argCost);  
};  
int Fishbread::GetCost()  
{  
    return cost;  
}  
void Fishbread::SetCost(int argCost)  
{  
    cost=argCost;  
}  
int main()  
{  
    Fishbread fish1;  
    fish1.SetCost(800);  
    cout<<"가격  
은?"<<fish1.GetCost()<<endl;  
    return 0;  
}
```

C++ Basics & Data Types

- const 멤버 함수

멤버 함수를 const 선언 시,
해당 클래스의 모든 멤버 값 변경 불능



```
class Fishbread
{
    public:
    void SetCost() const;
    private:
    int cost;
};
void Fishbread::SetCost() const
{
    ...
    cost=500; //error
}
```

C++ Basics & Data Types

- 클래스 선언과 멤버 함수 작성
 - 클래스와 사용자 간의 통신 인터페이스
 - 클래스의 자료형, 함수 종류 알림
 - *.h 파일 사용
- 함수 정의
 - 함수의 구체적 동작 정의
 - *.cpp 파일 사용

```
/*Fishbread.h*/
#include <iostream>
class Fishbread
{
public:
    Fishbread(int argCost,string argContent);
    ~Fishbread();
    int GetCost();
    void SetCost();

private:
    int cost;
    string content;
};
```

C++ Basics & Data Types

```
/*Fishbread.cpp*/
#include "Fishbread.h"
Fishbread::Fishbread()
{
}
Fishbread::Fishbread(int argCost,string argContent)
{
    cost=argCost;
    content=argContent;
}
Fishbread::~Fishbread()
{
    cout>>"붕어빵을 먹었습니다">>endl;
}
...
...
```

```
/*main.cpp*/
#include "Fishbread.hpp"
int main()
{
    Fishbread fish1(500,"팥");
    cout<<"가격은?"<<fish1.GetCost()<<endl;
    fish1.SetCost(800);
    cout<<"가격은?"<<fish1.GetCost()<<endl; return 0;
}
```


C++ Basics & Data Types

- 입출력
 - 출력 형태
 - Std::out << 출력대상
 - 개행
 - Std::endl;
 - 입력 형태
 - Std::cin>>'변수';

```
#include <iostream>
```

```
int main(void)
```

```
{
```

```
    int year = 2017;
```

```
    std::cout<<year<<"학년도 프로그래밍 수업"<<std::endl;
```

```
    std::cout<<"실습시간 입니다"<<std::endl;
```

```
    return 0;
```

```
}
```

C++ Basics & Data Types

- Data types
 - long double
 - double
 - float
 - unsigned long int (synonymous with unsigned long)
 - long int (synonymous with long)
 - unsigned int (synonymous with unsigned)
 - int
 - unsigned short int (synonymous with unsigned short)
 - short int (synonymous with short)
 - unsigned char
 - char
 - bool

C++ Basics & Data Types

- 문자형 char
 - `char A = 'A';`
- 정수형 int
 - `int A = 10;`
- 실수형 float, double
 - `float A = 12.34;`
- bool형 : true / false 가리키는 데이터형
 - `bool A = 0;`
 - `bool A = false;`
- void형 : 비어있음
- 열거형 enum : 사용자정의 데이터의 나열
 - `enum season { SPRING, SUMMER, FALL, WINTER }`

C++ Basics & Data Types

- 문자형 string : 문자의 모음
 - string A = "hello";
- 배열 : 같은 타입의 데이터의 집합
 - int A[200] = {1, 2, 3}
- 포인터 *& : 데이터의 주소를 저장하는 변수
 - int *A // 포인터 변수 생성
 - A = &B; //포인터 변수 A에 B의 주소 저장
 - *A = 100; (*A의 주소에 값 100을 저장)

Exercise 2 - 1

- Get two integers and two strings, print the **swapped** result.

```
#include <iostream>

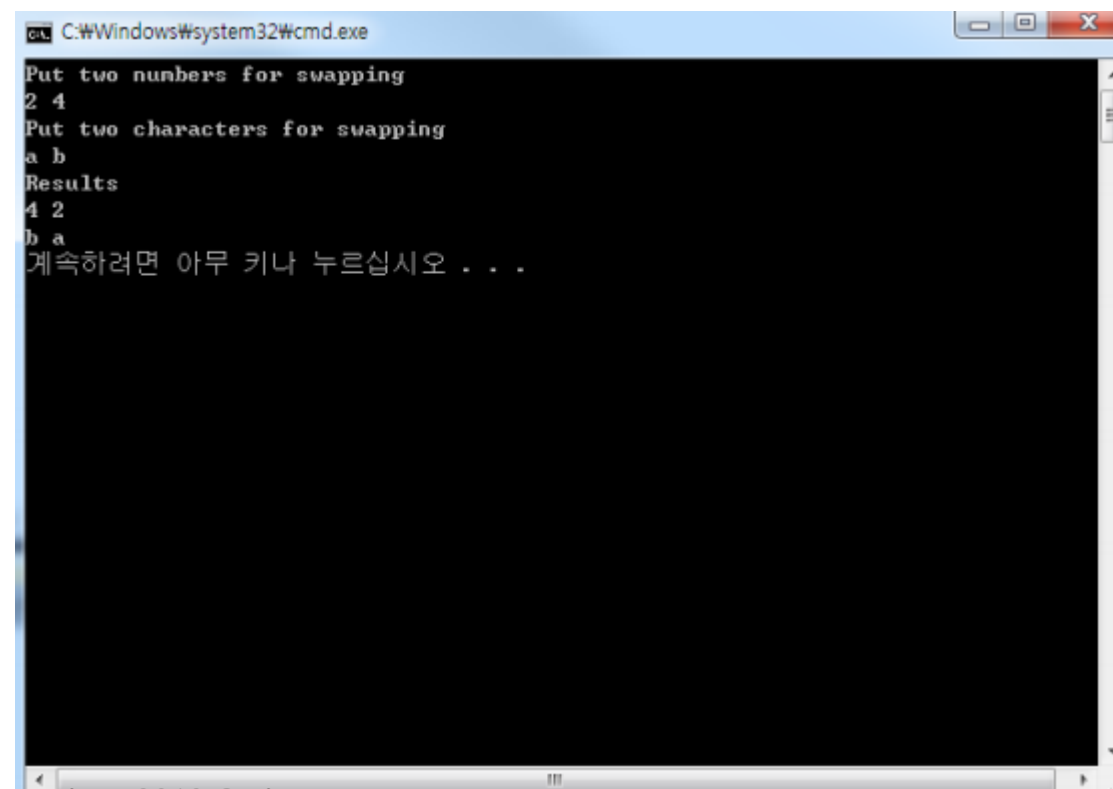
using std::cout;
using std::endl;
using std::cin;

void swap(int *a, int *b);
void swap(char *a, char *b);

int main(void)
{
    // FILL IN
}

void swap(int *a, int *b)
{
    // FILL IN
}

void swap(char *a, char *b)
{
    // FILL IN
}
```



```
C:\Windows\system32\cmd.exe
Put two numbers for swapping
2 4
Put two characters for swapping
a b
Results
4 2
b a
계속하려면 아무 키나 누르십시오 . . .
```

Exercise 2 - 2

- Get a character. Swap an upper case to a lower case, Swap a lower case to an upper case. And print out swapped result.
- HINT : ASCII code : A = 65 / a= 97

```
#include <iostream>

using namespace std;

int main(void)
{
    char alphabet;
    bool out = false; // terminate when out == true

    //FILL IN

    return 0;
}
```

```
Enter Capital or Small letter(0 for exit):a
input: a output: A
Enter Capital or Small letter(0 for exit):B
input: B output: b
Enter Capital or Small letter(0 for exit):c
input: c output: C
Enter Capital or Small letter(0 for exit):?
Enter character
Enter Capital or Small letter(0 for exit):0
exiting...
계속하려면 아무 키나 누르십시오 . . .
```