Lab 2

Introduction of Programming Environment
Basic of C++

• scan and print
• `cout << 1 << ‘a’ << “String” << endl;`
• `cin >> val1 >> val2;`
Function Overloading

```cpp
#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
#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;
}
In-line function

```cpp
#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
Exercise 2 - 1

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

```cpp
#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
}
```
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

```cpp
#include <iostream>
using namespace std;

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

    //FILL IN

    return 0;
}
```


C++ Basics & Data Types

• C++: Provide object oriented concept on software development platform

• OOP (object oriented programming): Think as one of object rather than considering data and the way to treat those data
C++ Basics & Data Types

• Class – as a new form
  • Memory size that a variable use
  • Information that a variable can have
  • Manipulations that a variable can be controlled by

• Definition of Class
  • A new form that combines all variables and related functions
  • Define a new class to create a new form
  • Similar to the “structure concept”, but the difference is that classes have functions also
  • Class = (Structure) + Function
C++ Basics & Data Types

• Vocabulary :
  • Member variable : Variables within the class
  • Member function, methods : Functions within the class, define what the object does
  • Object : Variable created by the class
C++ Basics & Data Types

- Define Class
- class [Class Name]
{
    Member variable;
    Member function;
};

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

Memory allocations occur when objects are created
C++ Basics & Data Types

- Define object
  Ex)
  Fishbread fish1("팥");
  Fishbread *fish2=new Fishbread("슈크림");

- Access to the object members
  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 만들어졌으며"
          <<fish1.roasting<<"정도로 구워짐\\n";
  return 0;
}
C++ Basics & Data Types

• Access control indicator
  • public: always accessible
  • Protected: when in inheritance relation, access granted on derived classes
  • Private: Access granted only within the class
  • Default: public

• Private: Encapsulation. It is used to protect class private values from outside. To access those private values, use public declared methods.
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;
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<<"가격은?"<<fish1.GetCost()<<endl;
    return 0;
}
C++ Basics & Data Types

- const member function

  If you define member function const, All member value within the class cannot be changed.

  ```cpp
class Fishbread
{
  public:
  void SetCost() const;
  private:
  int cost;
};
void Fishbread::SetCost() const
{
  ...
  cost=500; //error
}
```
C++ Basics & Data Types

• Declare class & Member function
  • Comm. Interface between class and the user
  • Notification of data type and kinds of function
  • Using *.h file

• Function Declaration
  • Define detailed actions of the function
  • Using *.cpp file

/*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

• Input/Output
  • Output
    • Std::out << [output]
  • New Line
    • Std::endl;
  • Input
    • Std::cin >> [variable];

#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

• Character datatype char
  • char A = ‘A’;

• Integer datatype int
  • int A = 10;

• Real number datatype float, double
  • float A = 12.34;

• Bool : true / false datatype
  • bool A = 0;
  • bool A = false;

• void : empty

• enum : A list of user-defined data
  • enum season { SPRING, SUMMER, FALL, WINTER }
C++ Basics & Data Types

• Text datatype string
  • string A = “hello”;

• Pointer * & : Variable that save the address of the data
  • int *A
  • A = &B;
  • *A = 100

• Array : A set of data of same datatype
  • int A[200] = {1, 2, 3}