Chapte01 C언어 기반의 C++ 1

* 1. printf와 scanf를 대신하는 입출력 방식

1)문자열”Hello world”의 출력

* 헤더파일 선언문 #include<iostream>
* std::cout과 <<을 이용한 출력
* std::endl을 이용한 개행

ex)

#include<iostream>

int main(void)

{

int num = 20;

std::cout<<’ ‘<<num<<”Hello “<<”World!”<<std::endl;

}

2)scanf를 대신하는 데이터의 입력

- std::cin과 >>을 이용한 출력

ex) std::cin>>num1>>val1;

* 1. 함수 오버로딩(Funtion Overlading)

1)함수의 오버로딩의 이해

함수의 호출 시 전달되는 인자의 종류나 숫자를 통해서 사용되는 함수를 구분 그래서 함수의 이름이 같더라도 사용이 가능

* 1. 매개변수의 디폴트 값(Default Value)

1)매개변수에 설정하는 ‘디폴트 값’의 의미

* 함수의 인자가 전달되는 공간에 디폴트 값을 설정 하는 방식으로 함수를 사용하는 곳에서 인자를 입력 하지 않으면 디폴트 값으로 사용하겠다는 의미

ex)

int MyFuncOne(int num=7)

{

return num+1;

}

int main(void)

{

MyFuncOne();

return = 0;

}

위의 main 함수를 보았을 때 MyFuncOne에 7이 입력된 것과 같은 형식

* 디폴트 값은 함수의 선언 부분에만 표현하면 됩니다.

ex)

#include<iostream>

int Adder(int num1=1, int num2=2)

* 부분적 디폴트 값 설정

함수를 정의하는 곳에서 인자를 적는 부분에서 오른쪽 매개변수의 디폴트 값을 비우는 형태로는 디폴트 값을 지정할 수 없다.

* 1. 인라인(inline) 함수

1)매크로 함수의 장점

* 매크로 함수의 장점은 빠른 실행 속도에 있다.
* 매크로 함수의 단점은 복잡한 함수의 형태를 정의하는데 한계가 있다.

2)C++기반의 인라인 함수 정의

* 인라인 함수 정의 방법

ex)

#include<iostream>

inline int SQUARE(int x)

{

return x \* x;

}

3)매크로 함수에는 있지만, 인라인 함수에는 없는 장점

* 자료형에 의하여 데이터 손실이 일어날 수 있다.
* 템플릿이라는 것을 활용 할 수 있지만 이것은 나중에 나오니 일단 스킵
  1. 이름공간(namespace)에 대한 소개

1)이름공간의 등장배경 및 기본원리

* 변수의 이름의 중복 사용
* 함수 앞에 ‘namespace’ 라는 것을 사용하여 이름공간 만들기
* ‘범위지정 연산자(scope resolution operator)’를 이용하여 이름공간 지정
* 이름공간 기반의 함수 선언과 정의의 구분

ex)

#include<iostream>

namespace BestComImpl

{

void SimpleFunc(void);

}

namespace BestComImpl

{

void PrettyFunc(void);

}

namespace ProgComImpl

{

void SimpleFunc(void);

}

int main(void)

{

BestComImpl::SimpleFunc();

return 0;

}

void BestComImpl::SimpleFunc(void)

{

std::cout << "BestCom이 정의한 함수" << std::endl;

PrettyFunc(); //동일 이름공간

ProgComImpl::SimpleFunc(); //다른 이름공간

}

void BestComImpl::PrettyFunc(void)

{

std::cout << "So Pretty!!" << std::endl;

}

void ProgComImpl::SimpleFunc(void)

{

std::cout << "ProgCom이 정의한 함수" << std::endl;

}

2)std::cout, std::cin, std::endl

* 위의 함수들은 이름공간 std에 선언된 것이라고 생각하고 ‘using std::cin’라고 선언하고 ‘std::cin’대신에 ‘cin’이라고 사용 가능
* 위의 함수들은 한꺼번에 한번만 선언하고 사용하고 싶은면 ‘using namespace std’라고 선언하고 사용 가능

3)별칭사용 가능

* 자주사용하는 이름의 루틴을 변수로 선언하여 사용하는 방식으로 ‘namespace ABC=AAA::BBB::CCC’ 와 같은 형식으로 선언을 하여 사용한다.