TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

목차

C++ 프로그래밍

1) constructor

생성자(constructor)

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

class A{

    public :
        A(void){
            cout << "A() Call" << endl;
        }
        A(int i){
            cout << "A(int i) Call" << endl;
        }
        A(const A& a){  // 객체 받을 때 사용법.
            cout << "A(const A & a) Call" << endl;
        }
};

int main(void){
    A obj1;
    A obj2(10);
    A obj3(obj2);

    return 0;
}
```

복사 생성자 (copy_constructor)

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

class Point{
    int x,y;

    public :
        Point(int _x, int _y){
            x = _x;
            y = _y;
        }
    void ShowData(void){
        cout << x << ' ' << y << endl;
    }
};

int main(void){

    Point p1(10, 20);
    Point p2(p1);

    p1.ShowData();
    p2.ShowData();
    return 0;
}
```

```
#include <iostream>
#include <string.h>
using namespace std;

class Person{
    char *name;
    char *phone;

public:
    Person(char *_name, char *_phone);
    Person(const Person& p);
    ~Person();
    void ShowData();
};

Person::Person(char *_name, char *_phone){

    name = new char[strlen(_name) + 1];
    strcpy(name, _name);

    phone = new char[strlen(_phone) + 1];
    strcpy(phone, _phone);
}

Person::~Person(void){  // destrcutor
    delete []name;
    delete []phone;
}

Person::Person(const Person& p){
    name = new char[strlen(p.name) + 1];
    strcpy(name, p.name);
    phone = new char[strlen(p.phone) + 1];
    strcpy(phone, p.phone);
}

void Person::ShowData(void){
    cout << "name : " << name << endl;
    cout << "phone : " << phone << endl;
}

int main(void){
    Person p1("Jo", "011-9272-6323");
    Person p2 = p1;

    p1.ShowData();
    p2.ShowData();
    return 0;
}
```

1) constructor

복사 생성자 (copy_constructor_method)

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

class A{

    int val;
public:
    A(int i){
        cout << "A(int i) Call" << endl;
        val = i;
    }
    A(const A& a){
        cout << "A(const A& a) Call" << endl;
        val = a.val;
    }
    void ShowData(void){
        cout << "val : " << val << endl;
    }
};

void function(A a){
    a.ShowData();
}

int main(void){
    A obj(30);
    function(obj);
    return 0;
}
```

복사 생성자 (copy_constructor_return)

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

class A{

    int val;
public:
    A(int i){
        cout << "A(int i) Call" << endl;
        val = i;
    }
    A(const A& a){
        cout << "A(const A& a) Call" << endl;
        val = a.val;
    }
    void ShowData(void){
        cout << "val : " << val << endl;
    }
};

A function(A& a){
    return a;
}

int main(void){

    A a(10);
    function(a).ShowData();
    return 0;
}
```

2) friend & template

friend

```cpp
#include <iostream>

using namespace std;

class A{
    private:
        int data;
        friend class B;
};

class B{

    public:
        void SetData(A& a, int data){
            a.data = data;
        }
};


int main(void){

    A a;
    B b;
    b.SetData(a, 10);
    return 0;
}
```

template

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

template <typename T>
T Add(T a, T b){
    return a+b;
}

int main(void){
    cout << Add(10,20) << endl;
    cout << Add(1.1, 2.2) << endl;
    return 0;
}
```

3) Inheritance & counter

```cpp
#include <iostream>
#include <string.h>

using namespace std;

class Person{

    int age;
    char name[20];
public:
    int GetAge(void) const{
        return age;
    }

    const char *GetName(void) const{  // const 함수 오버로딩
        return name;
    }

    Person(int _age = 1, char *_name = "noname"){
        age = _age;
        strcpy(name, _name);
    }
};

class Student: public Person{

    char major[20];
public:
    Student(char *_major){
        strcpy(major, _major);
    }
    const char *GetMajor(void) const{
        return major;
    }
    void ShowData(void) const{
        cout << "name : " << GetName() << endl;
        cout << "age : " << GetAge() << endl;
        cout << "major : " << GetMajor() << endl;
    }
};

int main(void){
    Student Park("Computer Science");
    Park.ShowData();
    return 0;
}
```

```cpp
#include <iostream>

using namespace std;

class Counter{

private:
    int val;
public:
    Counter(void){
        val = 0;
    }

    void Print(void){
        cout << val << endl;
    }
    friend void SetVal(Counter &c, int val);
};

void SetVal(Counter &c, int val){
    c.val = val;
}

int main(void){
    Counter cnt;
    cnt.Print();
    SetVal(cnt, 2002);
    cnt.Print();
    return 0;
}
```

4) Operator 1

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

class Point{

private:
    int x,y;
public:
    Point(int _x = 0, int _y = 0) : x(_x), y(_y) {} // :
    void ShowPosition(void);
    void operator + (int val);
};

void Point::ShowPosition(void){
    cout << x << " " << y <<endl;
}

void Point::operator+(int val){
    x += val;
    y += val;
}

int main(void){

    Point p(3,4);
    p.ShowPosition();

    p.operator + (10);
    p.ShowPosition();
    return 0;
}
```

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

class Point{
    private:
        int x,y;
    public:
        Point(int _x = 0, int _y = 0) : x(_x), y(_y) {}
        void ShowPosition(void);
        Point operator + (const Point& p);
};

void Point::ShowPosition(void){
    cout << x << " " << y << endl;
}

Point Point::operator + (const Point& p){
    Point temp(x + p.x, y + p.y);
    return temp;
}

int main(void){
    Point p1(1,2);
    Point p2(3,7);
    Point p3 = p1 + p2;
    p3.ShowPosition();
    return 0;
}
```

4) Operator 2

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

class Point{
private:
    int x,y;
public:
    Point(int _x = 0, int _y = 0) : x(_x), y(_y) {}
    void ShowPosition(void);
    Point& operator++(void);  // (void) 전위 연산    (int) 후위
    friend Point& operator--(Point& p);
};

void Point::ShowPosition(void){
    cout << x << " " << y << endl;
}

Point& Point::operator++(void){
    x++;
    y++;
    return *this;
}

Point& operator--(Point& p){
    p.x--;
    p.y--;
    return p;
}

int main(void){

    Point p(3,7);
    ++p;
    p.ShowPosition();

    --p;
    p.ShowPosition();

    ++(++p);
    p.ShowPosition();


    --(--p);
    p.ShowPosition();

    return 0;
}
```

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

class Point{
private:
    int x,y;
public:
    Point(int _x = 0, int _y = 0) : x(_x), y(_y) {}
    void ShowPosition(void);
    Point& operator++(void);  // (void) 전위 연산    (int) 후위
    Point& operator++(int);
};

void Point::ShowPosition(void){
    cout << x << " " << y << endl;
}

Point& Point::operator++(void){
    x++;
    y++;
    return *this;
}

Point& Point::operator++(int){
//  Point temp(x,y);
    // ++(*this);    요약하면 이것도 가능
    x++;
    y++;
    return *this;
}

int main(void){

    Point p1(3,7);
    (p1++).ShowPosition();
    p1.ShowPosition();

    Point p2(33,77);
    (p2++).ShowPosition();
    p2.ShowPosition();
    return 0;
}
```

4) Operator 3

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

class Point{
    private:
        int x,y;
    public:
        Point(int _x = 0, int _y = 0) : x(_x), y(_y) {}
        void ShowPosition(void);
        Point operator + (int val);
};

void Point::ShowPosition(void){
    cout << x << " " << y << endl;
}

Point Point::operator + (int val){
    Point temp( x +  val, y + val);
    return temp;
}

int main(void){

    Point p1(3,7);
    Point p2 = p1 + 3;
    p2.ShowPosition();
    return 0;
}
```

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

class Point{
    private:
        int x,y;
    public:
        Point(int _x = 0, int _y = 0) : x(_x), y(_y) {}
        void ShowPosition(void);
        Point operator + (int val);
        friend Point operator + (int val, Point& p);
};

void Point::ShowPosition(void){
    cout << x << " " << y << endl;
}

Point Point::operator + (int val){
    Point temp( x +  val, y + val);
    return temp;
}

Point operator + (int val, Point & p){
    return p + val;
}

int main(void){

    Point p1(3,7);
    Point p2 = p1 + 3;
    p2.ShowPosition();

    Point p3 = 7 + p2;
    p3.ShowPosition();

    return 0;
}
```