# 프로젝트 1
## (문제)

주어진 클래스를 기반으로 main 함수의 결과로써 오른편 "실행결과"가 나오도록 작성하시오.

[실행결과]
학생정보 입력: (ID name addr)
100 choi Seoul
same

```
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <fstream>
using namespace std;

class Student {
private:
        int id;
        char name[20];
        char addr[100];
};
```

# 프로젝트 1
## (문제)

주어진 클래스를 기반으로 main 함수의 결과로써 오른편 "실행결과"가 나오도록 작성하시오.

[실행결과]
학생정보 입력: (ID name addr)
100 choi Seoul
same

```
int main(void) {
        ofstream oFile("mydata.dat");
        Student st1, st2;
        cout << "학생정보 입력: (ID name addr) " << endl;
        cin >> st1;
        oFile << st1;
        oFile.close();

        ifstream iFile("mydata.dat");
        iFile >> st2;
        if (st1 == st2)
                cout << "same" << endl;
        else
                cout << "diff" << endl;

        return 0;
}
```

# 프로젝트 1
## (풀이)

```
class Student {
private:
        int id;
        char name[20];
        char addr[100];
public:
        Student() {
                id = 0;
                memset(name, 0x00, sizeof(name));
                memset(addr, 0x00, sizeof(addr));
        }
        int GetId() const { return id; };
        const char* GetName() const { return name; }
        const char* GetAddr() const { return addr; }
        void SetId(int _id) { id = _id; }
        void SetName(const char* _name) { strcpy(name, _name); }
        void SetAddr(const char* _addr) { strcpy(name, _addr); }
        bool operator==(const Student& st) const {
            if (id == st.id && strcmp(name, st.name) && strcmp(addr, st.addr))
                return true;
            return false;
        }
        friend ostream& operator<<(ostream& os, const Student& st);
        friend istream& operator>>(istream& is, Student& st);
};
```

# 프로젝트 1
## (풀이)

```
ostream& operator<<(ostream& os, const Student& st) {
        os << st.id << st.name << st.addr;
        return os;
}

istream& operator>>(istream& is, Student& st) {
        is >> st.id >> st.name >> st.addr;
        return is;
}
```

아래 main 함수의 결과로써 오른편 "실행결과"가 나오도록 최소한의 함수만 구현한 클래스를 만드시오. 단, 클래스는 1개만 작성하며 동적 배열을 할당 받아 구현하도록 한다.

```
#include <iostream>
#include <fstream>
using namespace std;
int main(void) {
        BoundCheckArray<int> iarr(5);
        BoundCheckArray<double> farr(5);
        for (int i = 0; i < 5; i++) {
                iarr[i] = i + 1;
                farr[i] = i + 1 + 0.5;
        }
        for (int i = 0; i < 5; i++) {
                cout << iarr[i] << ", ";
                cout << farr[i] << endl;
        }
        BoundCheckArray<int> iarr2(iarr);
        BoundCheckArray<double> farr2;
        farr2 = farr;
        for (int i = 0; i < 5; i++) {
                cout << iarr2[i] << ", ";
                cout << farr2[i] << endl;
        }
        return 0;
}
```

[실행결과]

```
1, 1.5
2, 2.5
3, 3.5
4, 4.5
5, 5.5
1, 1.5
2, 2.5
3, 3.5
4, 4.5
5, 5.5
```

# 프로젝트 2

# 프로젝트 2
## (풀이)

```
template <typename T>
class BoundCheckArray {
private:
        T* arr;
        int arrlen;
public:
        BoundCheckArray(int len = 0);
        BoundCheckArray(const BoundCheckArray& data);
        BoundCheckArray<T>& operator=(const BoundCheckArray<T>&
data);
        T& operator[] (int idx);
        T operator[] (int idx) const;
        ~BoundCheckArray();
};
```

# 프로젝트 2
## (풀이)

```
template <typename T>
BoundCheckArray<T>::BoundCheckArray(const BoundCheckArray& data) {
        if (data.arrlen == 0) {
                arrlen = 0;
                arr = NULL;
        }
        else {
                arrlen = data.arrlen;
                arr = new T[arrlen];
                for (int i = 0; i < arrlen; i++)
                        arr[i] = data.arr[i];
        }
}
```

# 프로젝트 2
## (풀이)

```
template <typename T>
BoundCheckArray<T>& BoundCheckArray<T>::operator=(const
BoundCheckArray<T>& data) {
        if (arr)
                delete[] arr;

        if (data.arrlen == 0) {
                arrlen = 0;
                arr = NULL;
        }
        else {
                arrlen = data.arrlen;
                arr = new T[arrlen];
                for (int i = 0; i < arrlen; i++)
                        arr[i] = data.arr[i];
        }
        return *this;
}
```

# 프로젝트 2
## (풀이)

```
template <typename T>
BoundCheckArray<T>::BoundCheckArray(int len) :arrlen(len)
{
        if (len > 0)
                arr = new T[len];
        else
                arr = NULL;
}

template <typename T>
T& BoundCheckArray<T>::operator[] (int idx)
{
        if (idx < 0 || idx >= arrlen)
        {
                cout << "Array index out of bound exception" << endl;
                exit(1);
        }
        return arr[idx];
}
```

# 프로젝트 2
## (풀이)

```
template <typename T>
T BoundCheckArray<T>::operator[] (int idx) const {
        if (idx < 0 || idx >= arrlen)
        {
                cout << "Array index out of bound exception" << endl;
                exit(1);
        }
        return arr[idx];
}

template <typename T>
BoundCheckArray<T>::~BoundCheckArray() {
        if (arr != NULL)
                delete[] arr;
}
```

# 프로젝트 3

아래 main 함수의 결과로써 오른편 "실행결과"가 나오도록 필요한 클래스들을 만드시오. 추상클래스 Figure 를 사용하도록 한다.

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

int main(void) {
	vector<Figure*> farr;
	farr.push_back(new Rect(100, 100, 200, 200));
	farr.push_back(new Line(100, 100, 200, 200));
	while (!farr.empty()) {
		farr.back()->ShowInfo();
		cout << "Area: " << farr.back()->GetArea() << endl;
		farr.pop_back();
	}

	return 0;
}
```

[실행결과]

```
Line: (100,100) ~ (200,200)
Area: 0
Rect: (100,100) ~ (200,200)
Area: 10000
```

# 프로젝트 3
## (풀이)

```
class Figure {
protected:
        int x1, y1, x2, y2;
public:
        Figure(int _x1, int _y1, int _x2, int _y2) {
                x1 = _x1; y1 = _y1;  x2 = _x2; y2 = _y2;
        }
        virtual void ShowInfo() const = 0;
        virtual int GetArea() const = 0;
};
```

# 프로젝트 3

## (풀이)

```
class Rect : public Figure {
public:
    Rect(int _x1, int _y1, int _x2, int _y2) : Figure(_x1, _y1, _x2, _y2){}
    virtual void ShowInfo() const {
        cout << "Rect: " << "(" << x1 << "," << y1 << ") ~ (" << x2 << ","
<< y2 << ")" << endl;
    }
    virtual int GetArea() const {
        return (x2 - x1) * (y2 - y1);
    }
};
class Line : public Figure {
public:
    Line(int _x1, int _y1, int _x2, int _y2) : Figure(_x1, _y1, _x2, _y2) {}
    virtual void ShowInfo() const {
        cout << "Line: " << "(" << x1 << "," << y1 << ") ~ (" << x2 << ","
<< y2 << ")" << endl;
    }
    virtual int GetArea() const {
        return 0;
    }
};
```