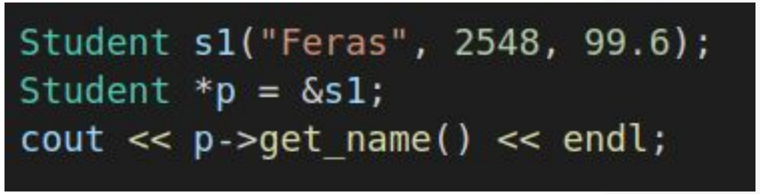
Topic 4-2

Pointers to, and Arrays of Objects

THE UNIVERSITY *of* ADELAIDE

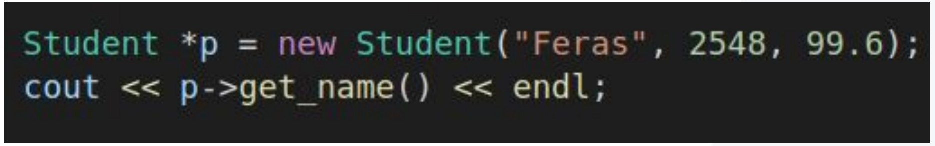
Pointer to an object

- We can define a pointer that contains the address of an object.



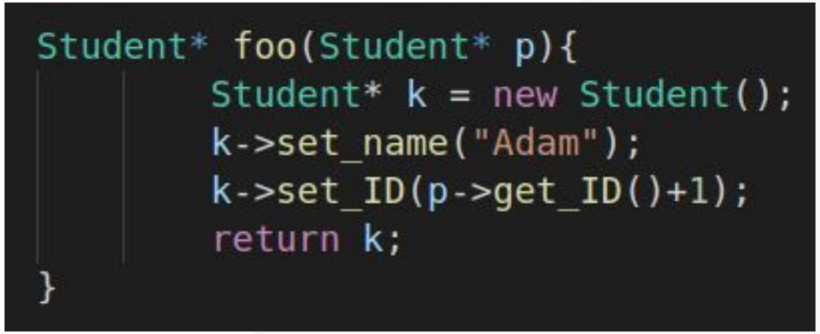
```cpp
Student s1("Feras", 2548, 99.6);
Student *p = &s1;
cout << p->get_name() << endl;
```

- We can allocate memory space for an object and store its address in a pointer.



```cpp
Student *p = new Student("Feras", 2548, 99.6);
cout << p->get_name() << endl;
```

- We can pass an address to an object to a function using a pointer.

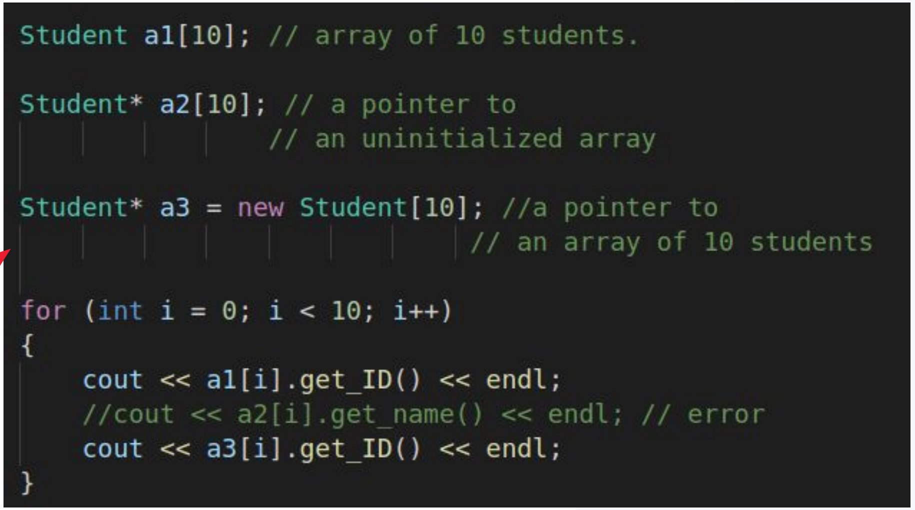


```cpp
Student* foo(Student* p){
        Student* k = new Student();
        k->set_name("Adam");
        k->set_ID(p->get_ID()+1);
        return k;
}
```

Array of Objects

- We can define an array of objects similar to an array of built-in data types like int.

- We can create a dynamic array of objects using “new”.

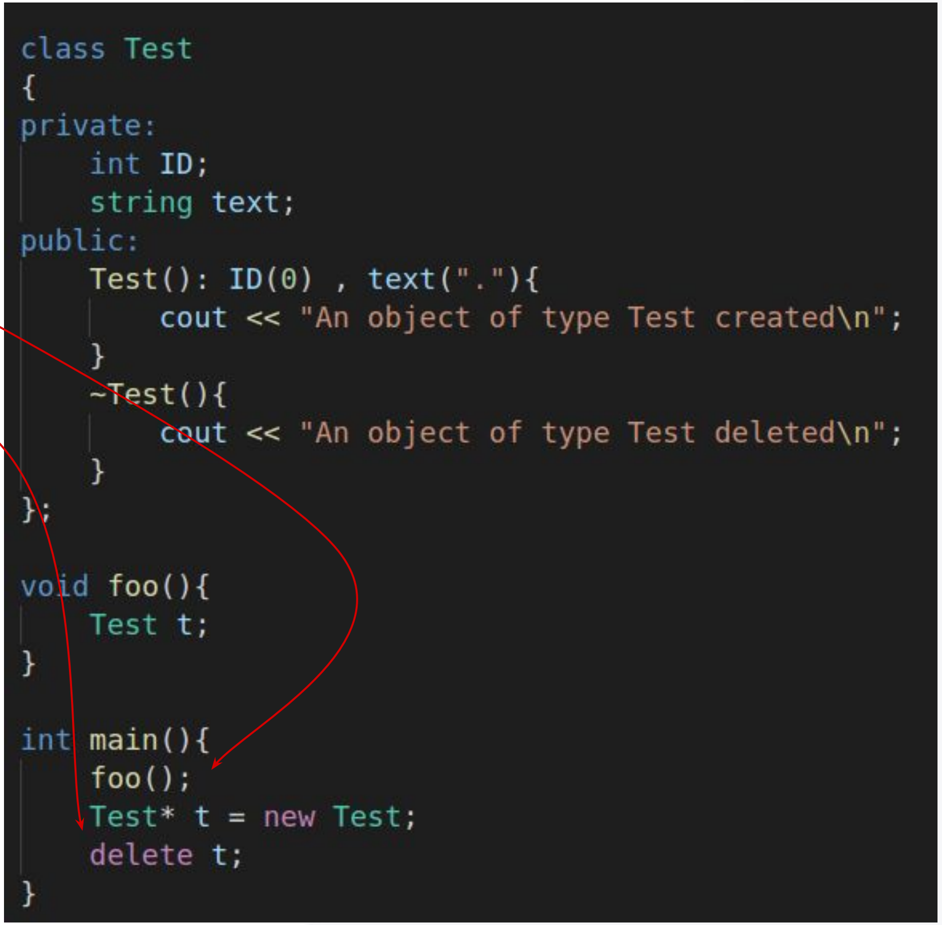


```
Student a1[10]; // array of 10 students.

Student* a2[10]; // a pointer to
                // an uninitialized array

Student* a3 = new Student[10]; //a pointer to
                               // an array of 10 students

for (int i = 0; i < 10; i++)
{
    cout << a1[i].get_ID() << endl;
    //cout << a2[i].get_name() << endl; // error
    cout << a3[i].get_ID() << endl;
}
```

Destructors

A destructor is a member function of a class that is called automatically when an object of the class goes out of scope or a pointer to the object is deleted.

Destructors are used to eliminate any dynamic variables that have been created by the object so that the memory occupied by these dynamic variables is returned to the freestore. Destructors may perform other cleanup tasks as well.

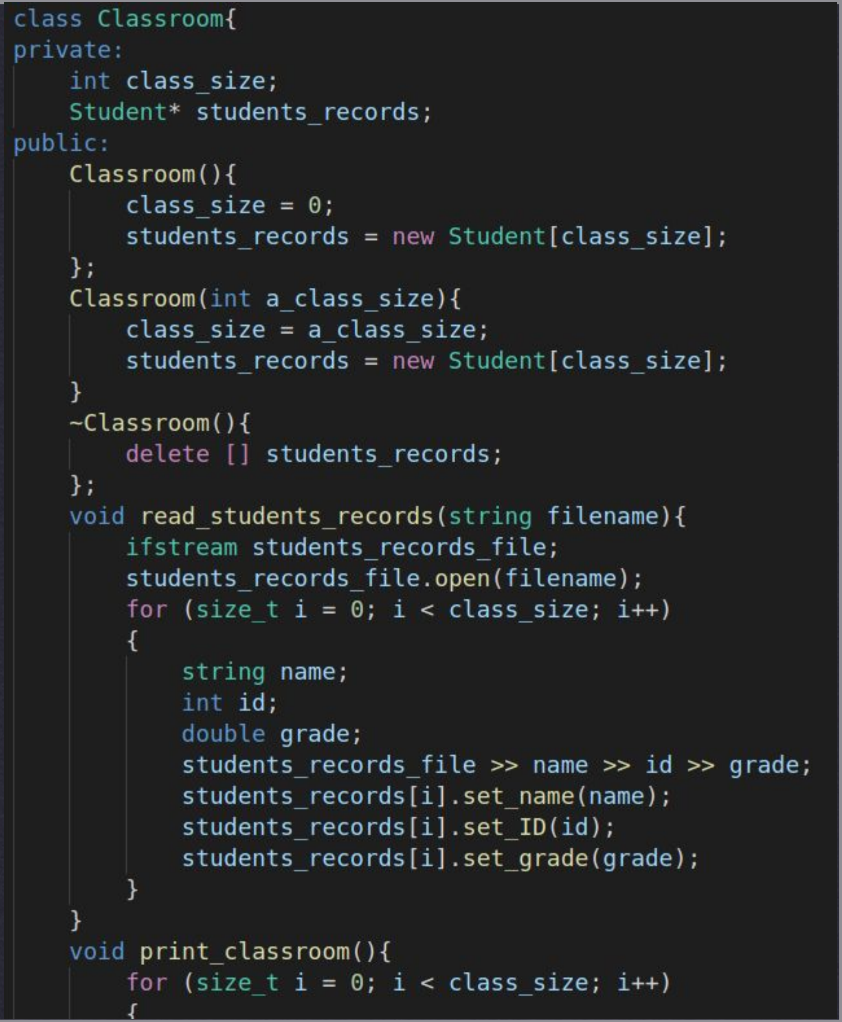


```cpp
class Test
{
private:
    int ID;
    string text;
public:
    Test(): ID(0) , text("."){
        cout << "An object of type Test created\n";
    }
    ~Test(){
        cout << "An object of type Test deleted\n";
    }
};

void foo(){
    Test t;
}

int main(){
    foo();
    Test* t = new Test;
    delete t;
}
```

Use Case

Write a program to manage the student's records in a classroom. Each student has a unique ID, name and grade. The program should be able to print all the students in the classroom, find the grade of a student based on their ID and update the grade of a student. The program should write the students' records to a file and also read the records from a file.

Demo 3



```cpp
class Classroom{
private:
    int class_size;
    Student* students_records;
public:
    Classroom(){
        class_size = 0;
        students_records = new Student[class_size];
    };
    Classroom(int a_class_size){
        class_size = a_class_size;
        students_records = new Student[class_size];
    }
    ~Classroom(){
        delete [] students_records;
    };
    void read_students_records(string filename){
        ifstream students_records_file;
        students_records_file.open(filename);
        for (size_t i = 0; i < class_size; i++)
        {
            string name;
            int id;
            double grade;
            students_records_file >> name >> id >> grade;
            students_records[i].set_name(name);
            students_records[i].set_ID(id);
            students_records[i].set_grade(grade);
        }
    }
    void print_classroom(){
        for (size_t i = 0; i < class_size; i++)
        {
```