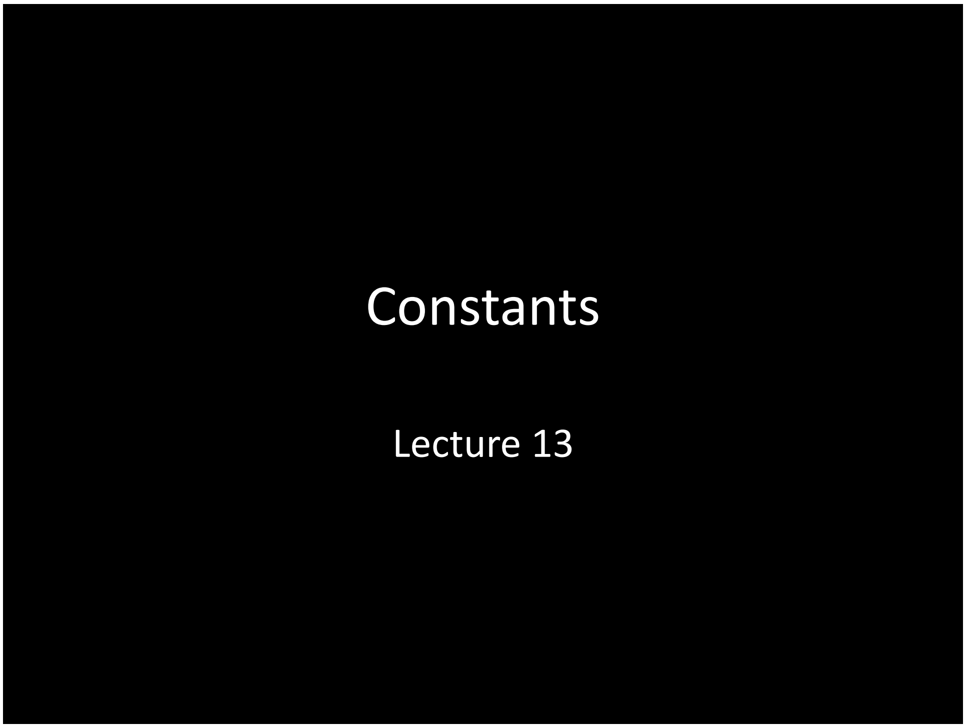
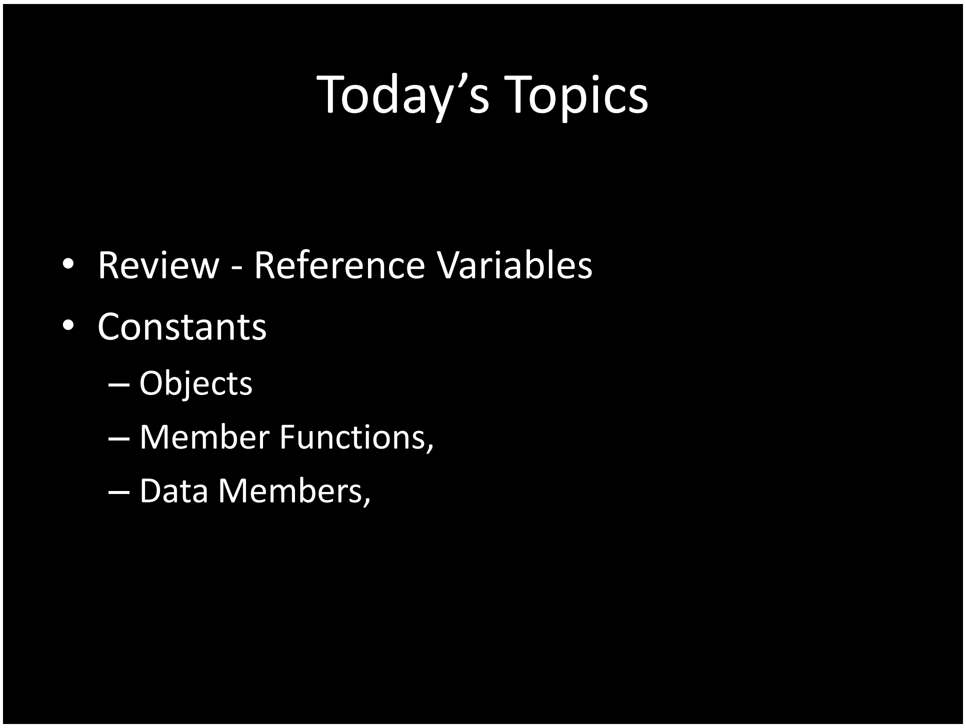
Week 05

- Constant Objects, Functions & Data Member,
- Static Member Variables & Static Functions



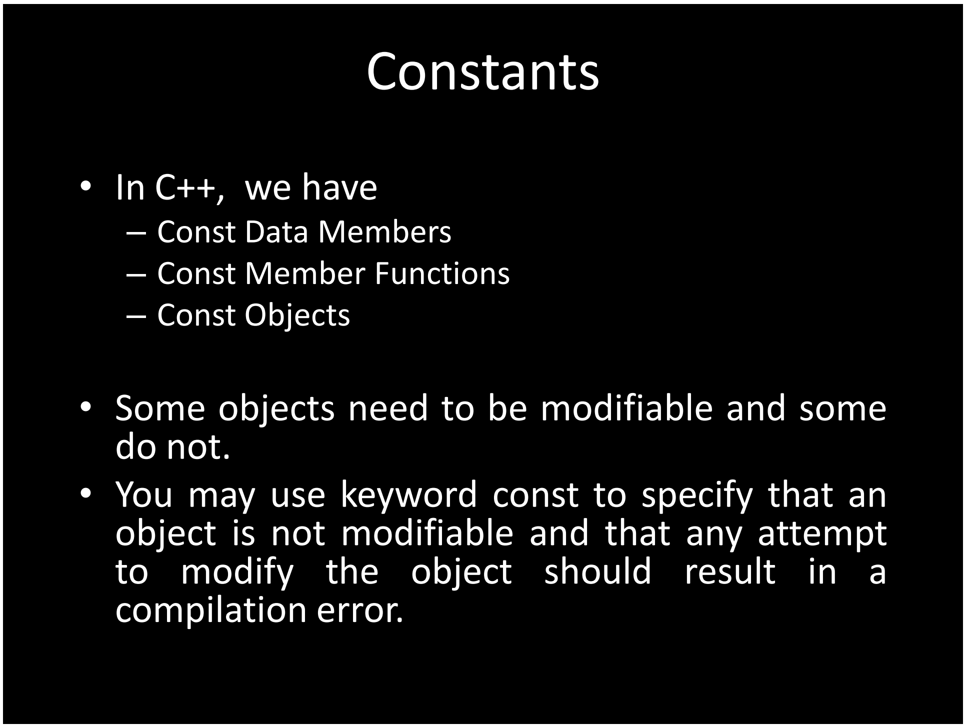
Constants

Lecture 13



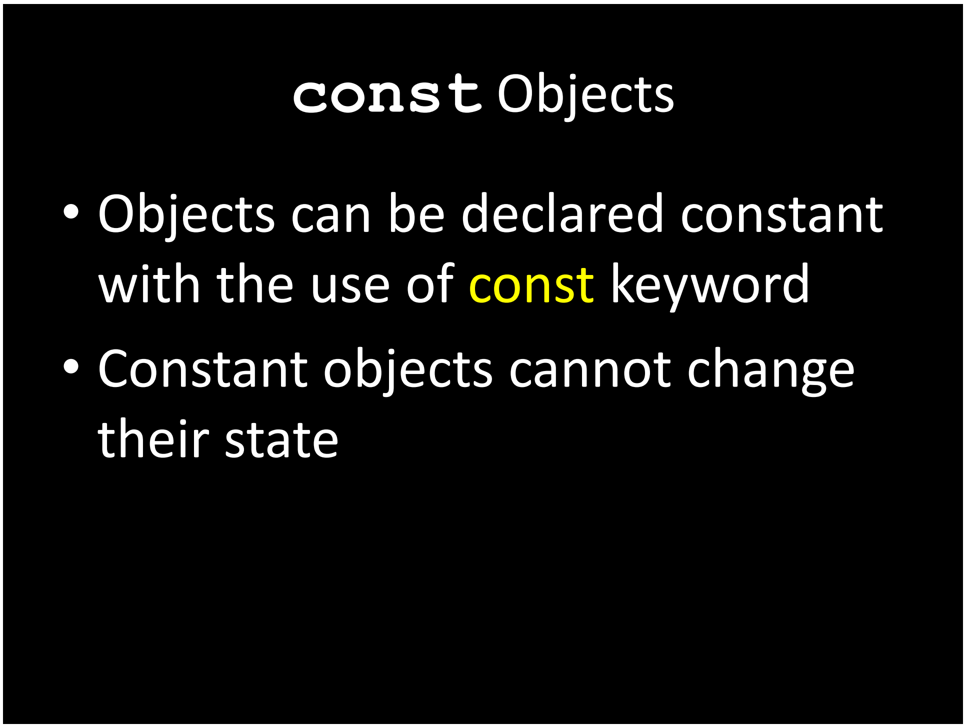
Today’s Topics

- Review - Reference Variables
- Constants
  – Objects
  – Member Functions,
  – Data Members,



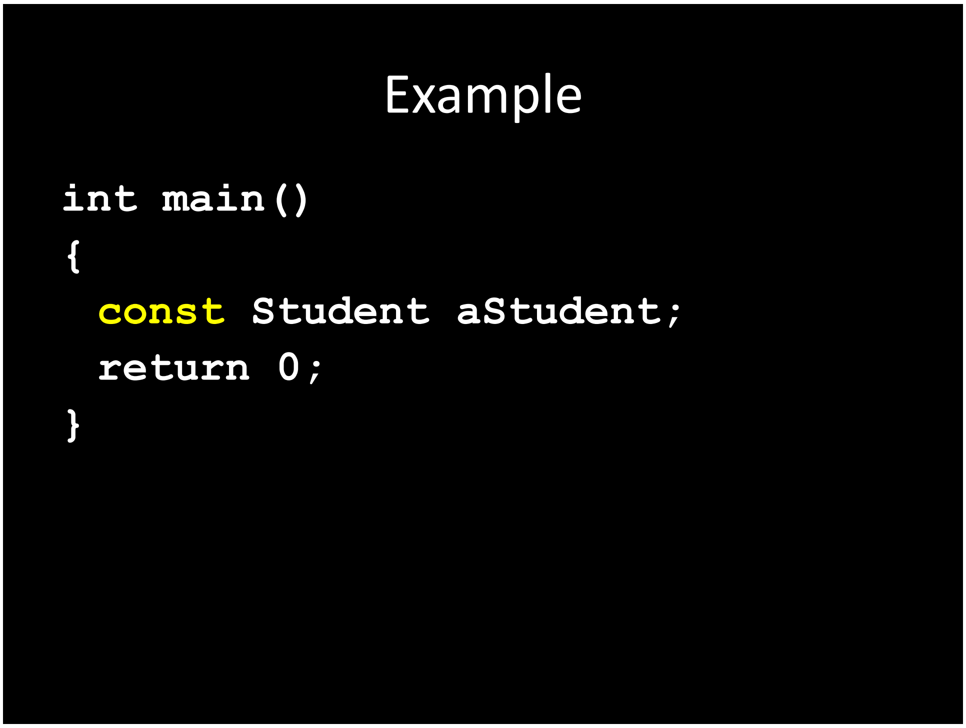
Constants

- In C++, we have
  – Const Data Members
  – Const Member Functions
  – Const Objects
- Some objects need to be modifiable and some do not.
- You may use keyword const to specify that an object is not modifiable and that any attempt to modify the object should result in a compilation error.



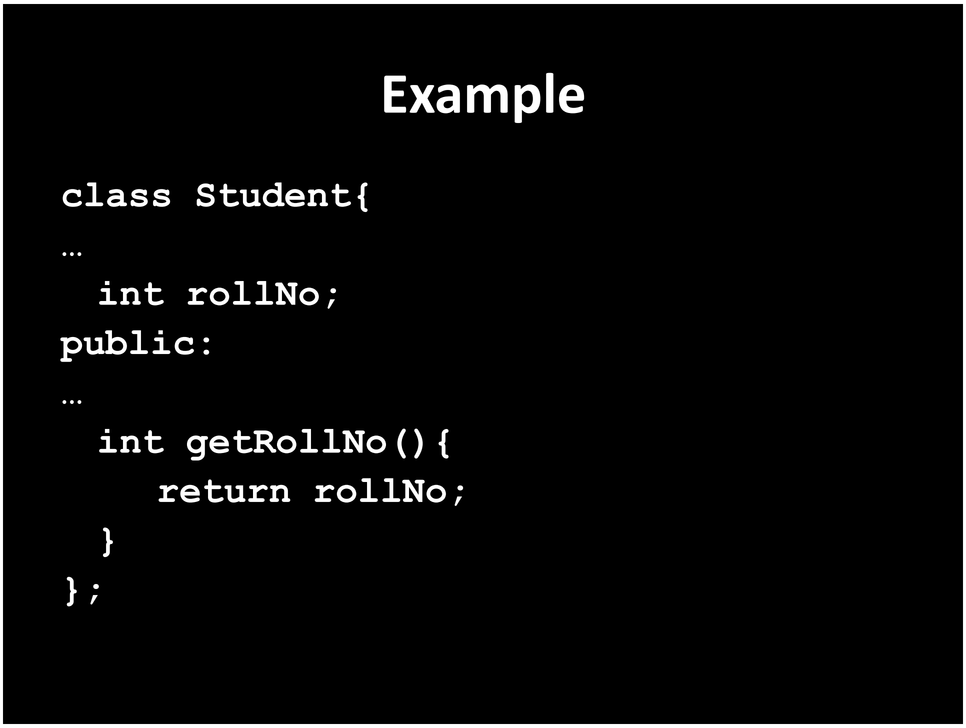
`const` Objects

- Objects can be declared constant with the use of const keyword
- Constant objects cannot change their state



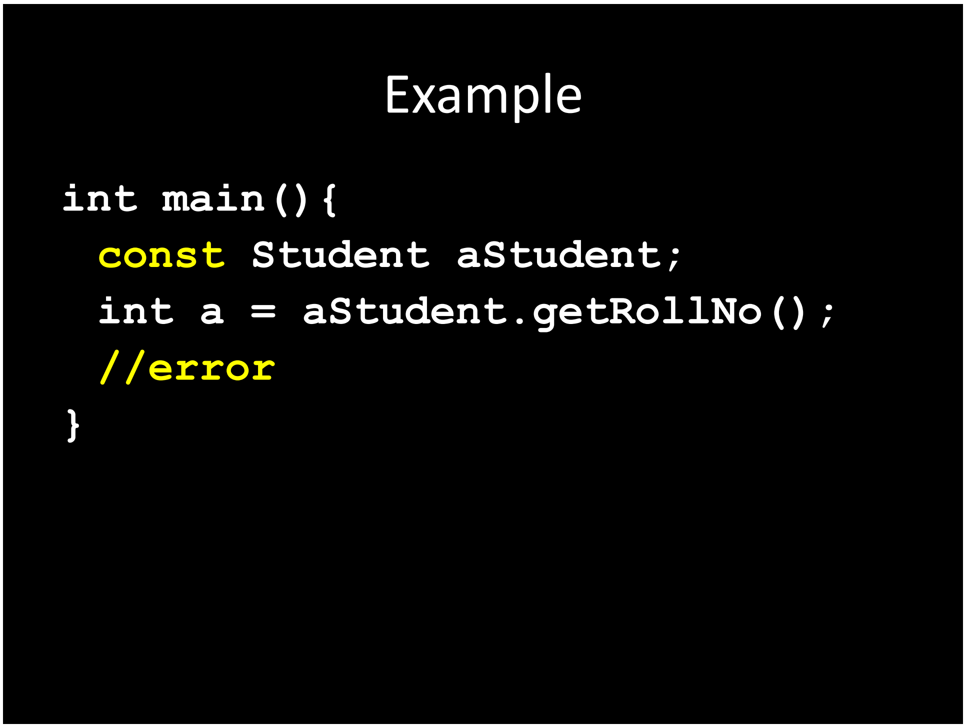
Example

```
int main()
{
  const Student aStudent;
  return 0;
}
```



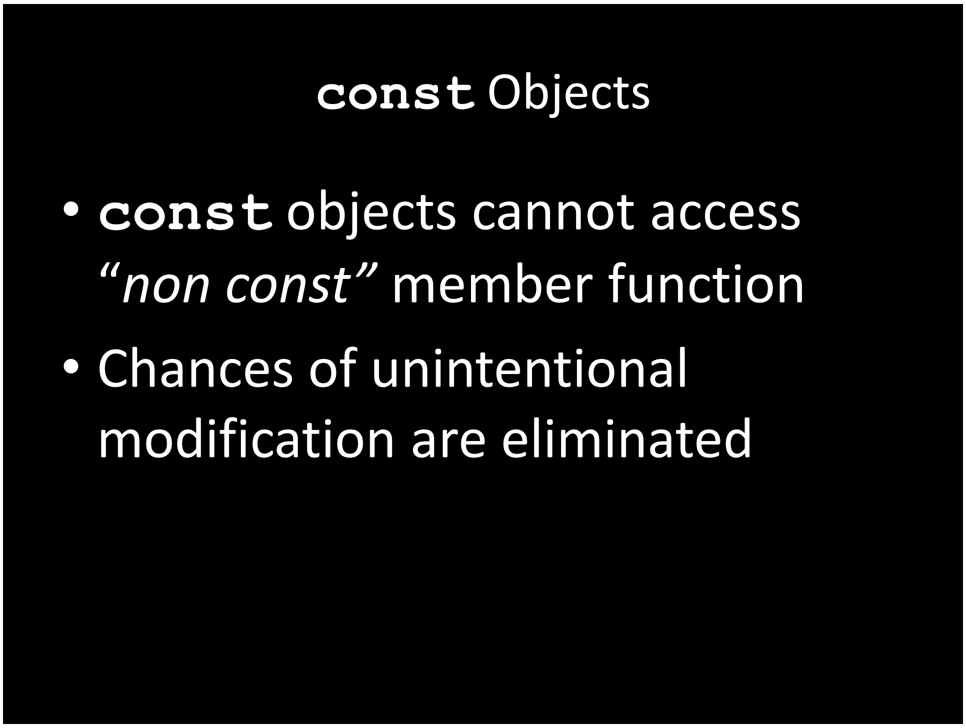
Example

```
class Student{
…
  int rollNo;
public:
…
  int getRollNo(){
      return rollNo;
  }
};
```



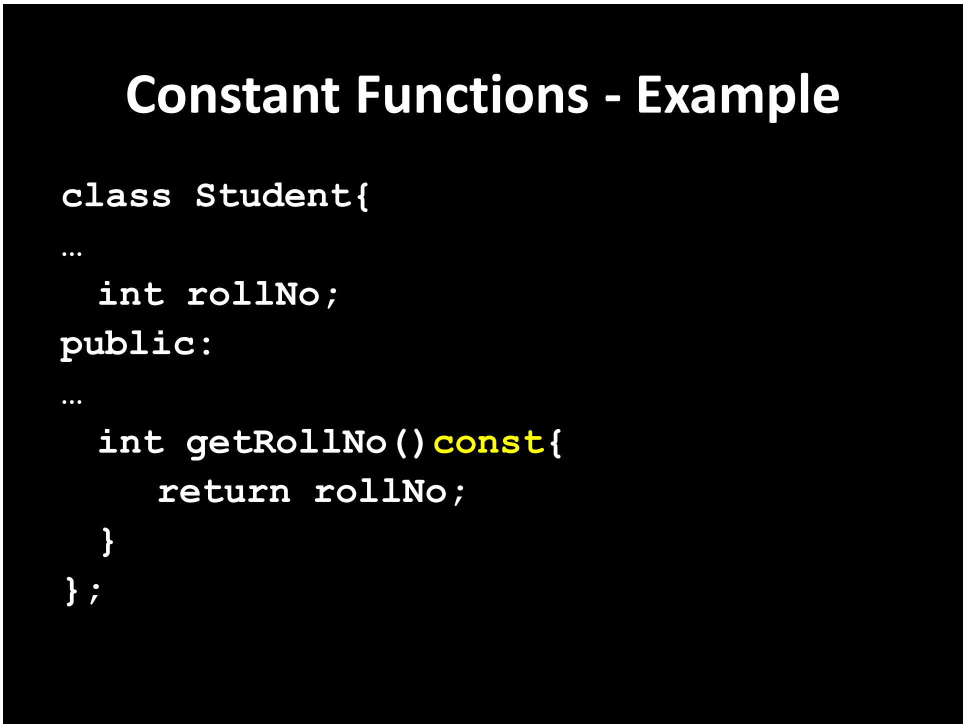
Example

```
int main(){
  const Student aStudent;
  int a = aStudent.getRollNo();
  //error
}
```



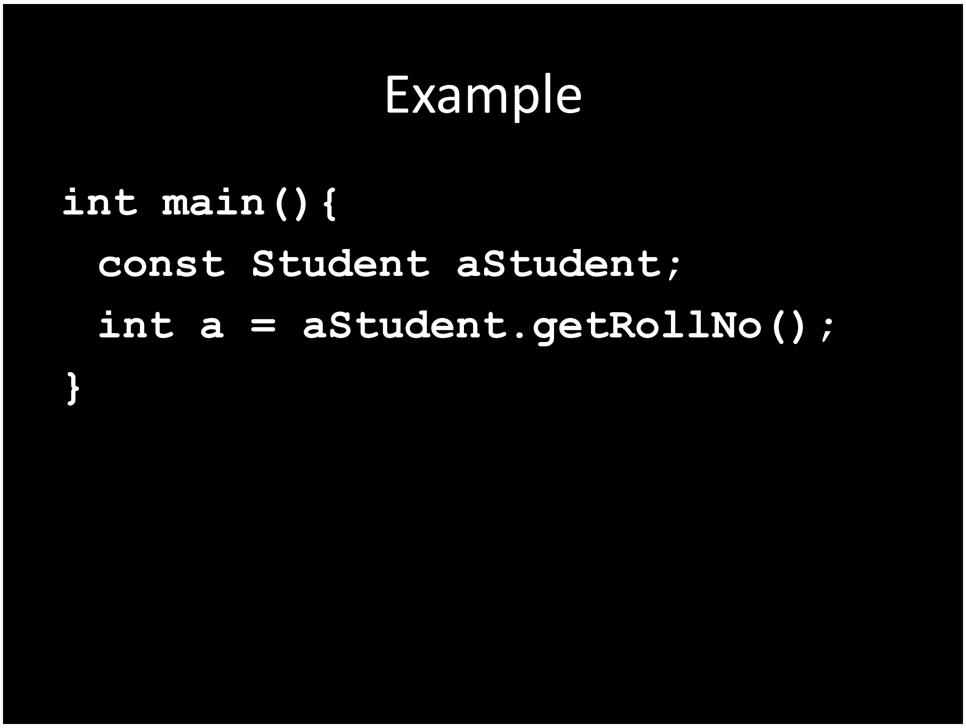
`const` Objects

- `const` objects cannot access *"non const"* member function
- Chances of unintentional modification are eliminated



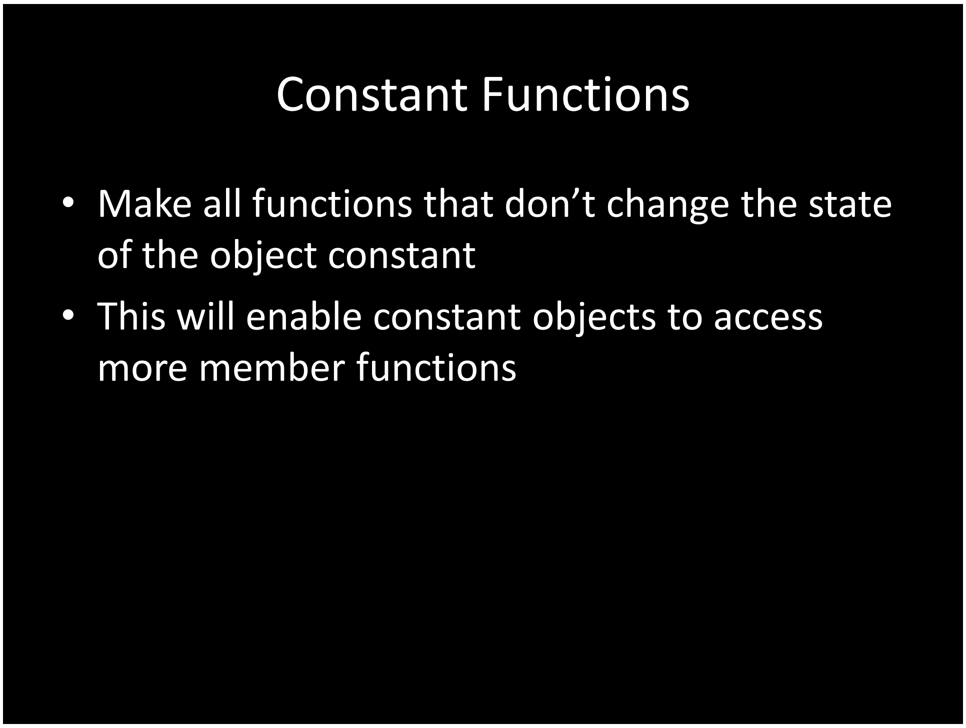
Constant Functions - Example

```
class Student{
…
  int rollNo;
public:
…
  int getRollNo()const{
     return rollNo;
  }
};
```



Example

```
int main(){
  const Student aStudent;
  int a = aStudent.getRollNo();
}
```



Constant Functions

- Make all functions that don’t change the state of the object constant
- This will enable constant objects to access more member functions

Const Data Members

- Const are those, that value are not changed or remain constant through out the program
- Intialize at the time of declaration only
- Never update or change value at any stage of program
- const int value1 = 5; // copy initialization

Problem

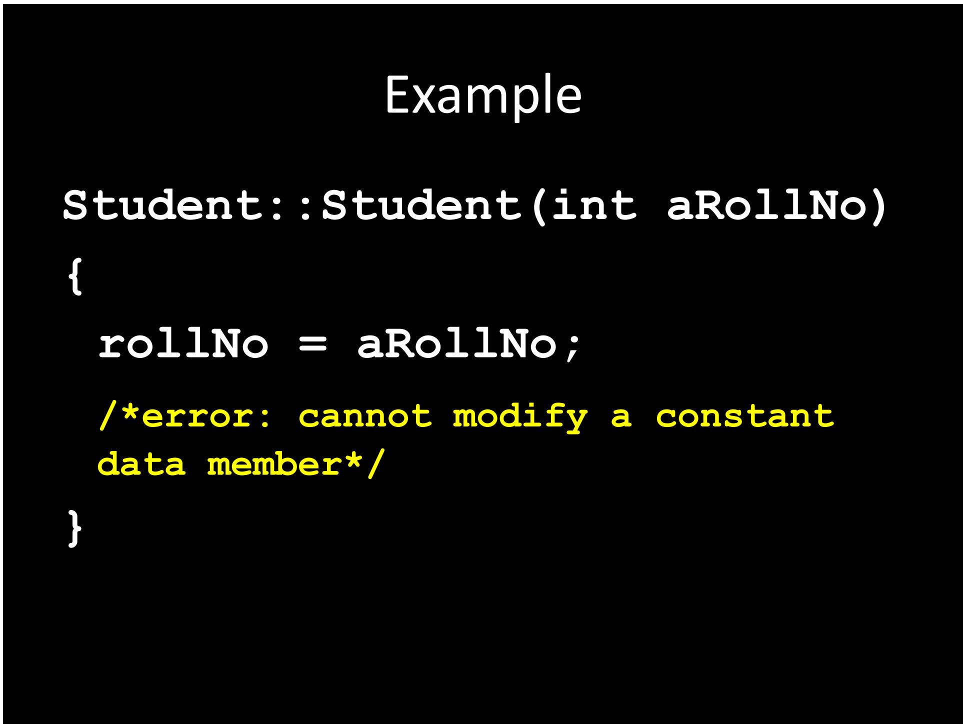
- Change the class Student such that a student is given a roll number when the object is created and cannot be changed afterwards

Student Class

```
class Student{
…
  int rollNo;
public:
  Student(int aNo);
  int getRollNo();
  void setRollNo(int aNo);
…
};
```

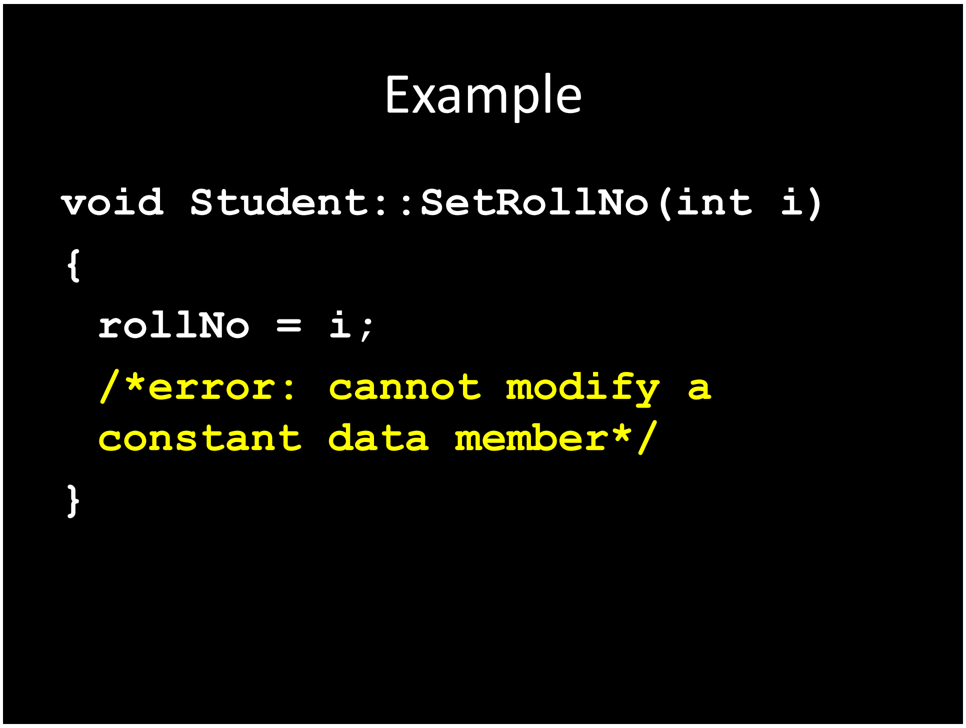
Modified Student Class

```
class Student{
…
  const int rollNo;
public:
  Student(int aNo);
  int getRollNo();
  void setRollNo(int aNo);
…
};
```



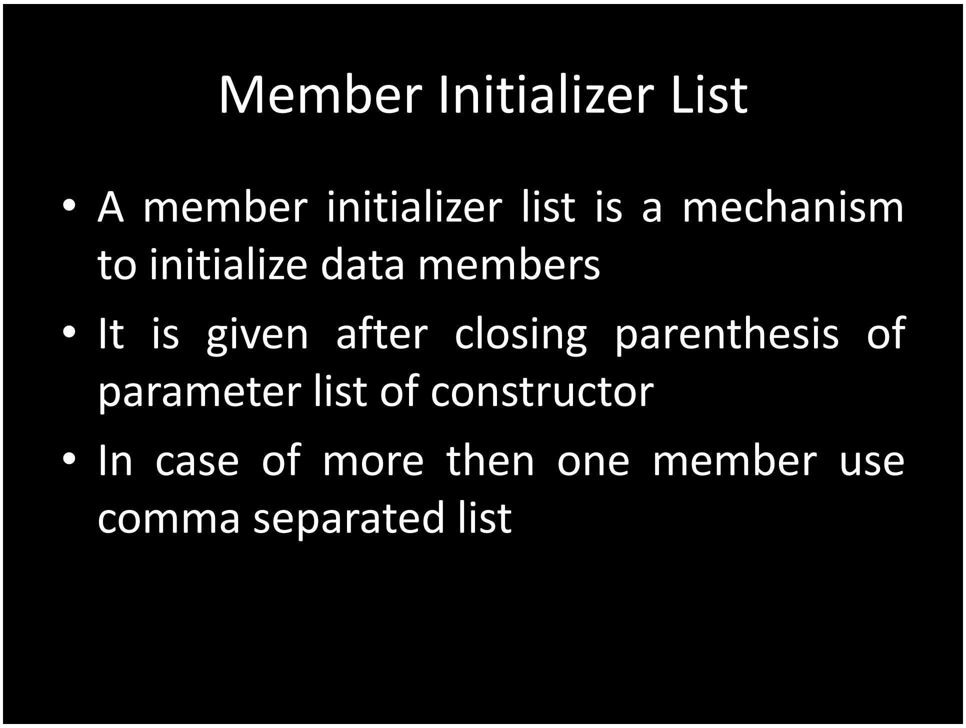
Example

```
Student::Student(int aRollNo)
{
  rollNo = aRollNo;
  /*error: cannot modify a constant
  data member*/
}
```



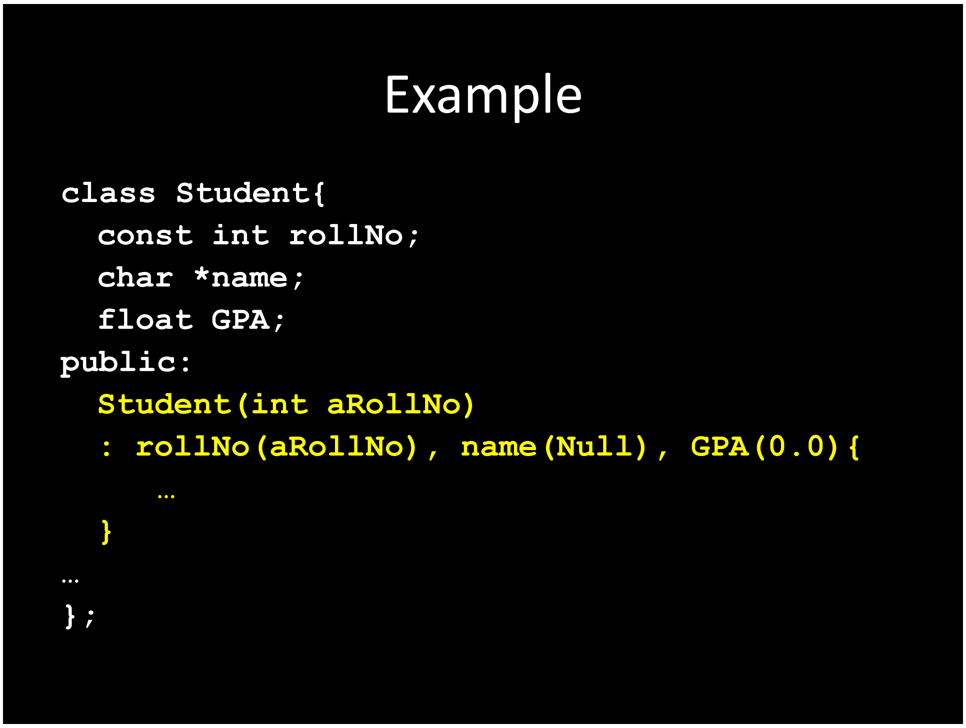
Example

```
void Student::SetRollNo(int i)
{
  rollNo = i;
  /*error: cannot modify a
  constant data member*/
}
```



Member Initializer List

- A member initializer list is a mechanism to initialize data members
- It is given after closing parenthesis of parameter list of constructor
- In case of more then one member use comma separated list



Example

```
class Student{
  const int rollNo;
  char *name;
  float GPA;
public:
  Student(int aRollNo)
  : rollNo(aRollNo), name(Null), GPA(0.0){
     …
  }
…
};
```

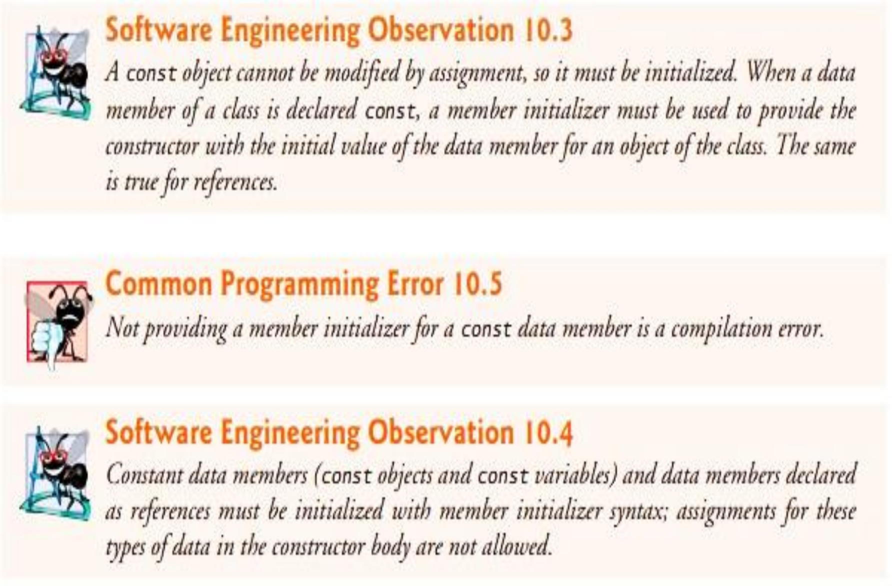
Performance Tip

- Passing an object by value is good from a security stand point, because the called function has no access to the original object in the caller, but pass-by-value can degrade performance when making a copy of a large object.

- An object can be passed by reference by passing either a pointer or a reference to the object. Pass-by-reference offers good performance but is weaker from a security standpoint, because the called function is given access to the original object.

- Pass-by-const-reference is a safe, good-performing alternative(this can be implemented with a const reference parameter or with a pointer-to-const-data parameter).

Good Programming Practice(s)



Software Engineering Observation 10.3

A `const` *object cannot be modified by assignment, so it must be initialized. When a data member of a class is declared* `const`*, a member initializer must be used to provide the constructor with the initial value of the data member for an object of the class. The same is true for references.*

Common Programming Error 10.5

Not providing a member initializer for a `const` *data member is a compilation error.*

Software Engineering Observation 10.4

Constant data members (`const` *objects and* `const` *variables) and data members declared as references must be initialized with member initializer syntax; assignments for these types of data in the constructor body are not allowed.*

Review