CSE 1201 & 1202
oop

30/04/25

Dr. Md. Toukir Ah

Department of Computer Science and Engineering, PUST

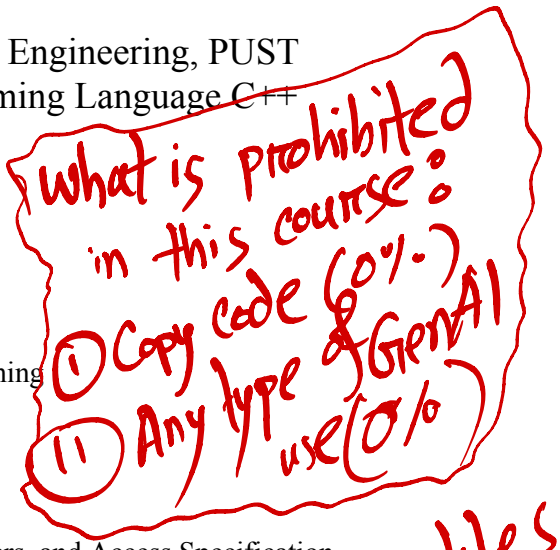
CSE 1201: Object-Oriented Programming Language C++

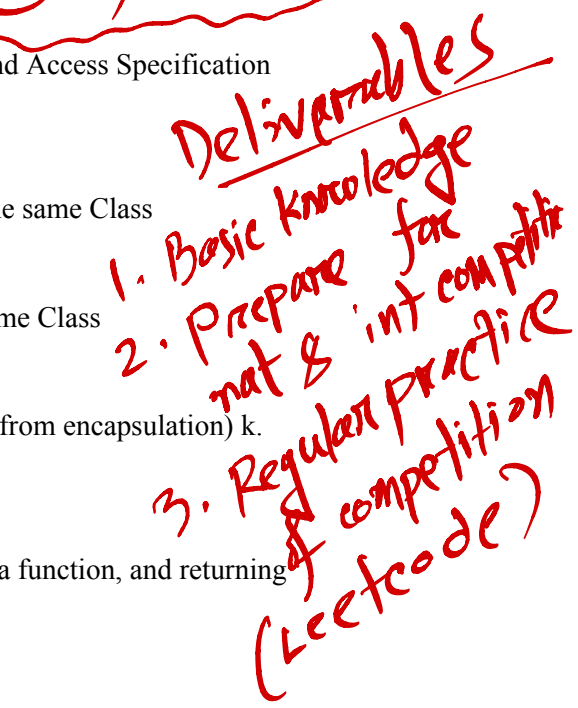
Course Outline

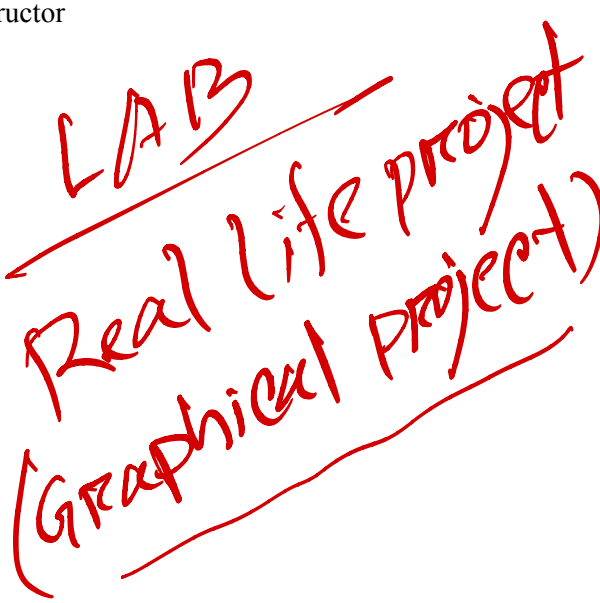
1. Brief history
2. Introduction
3. C++ Header Files and Namespace
4. C++ Console I/O (***cout*** and ***cin***)
5. Fundamental Concept in Object-Oriented Programming
    a. Encapsulation
    b. Polymorphism
    c. Inheritance
6. C++ Class and Object (Encapsulation)
    a. Class Definition, Data and Function Members, and Access Specification (private, public, and protected)
    b. Object from a Class
    c. Accessing Object Members
    d. Pointer from a Class and Pointing an Object of the same Class
    e. Accessing Object Members through the Pointer
    f. Constructor and Destructor functions
    g. Assigning one Object to another Object of the same Class
    h. Passing Object as the parameter to a function
    i. Returning an Object from a function
    j. Friend function (non-member function, deviation from encapsulation) k. Array of Objects
    l. Special pointer ***this***
    m. Special operators: ***new*** and ***delete***
    n. Reference, passing reference as the parameter to a function, and returning reference from a function
7. C++ Function Overloading (Polymorphism)
    a. Non-member function overloading
    b. Member function overloading
    c. Constructor function overloading and copy constructor
8. C++ Operator Overloading (Polymorphism)
    a. Overloading binary operator
    b. Overloading unary operator
    c. Using friend function to overload operator
9. C++ Inheritance
    a. Base and Derived Classes
    b. Access control (private, protected, and public)
    c. Multiple Inheritance
    d. Virtual Base Class
    e. Virtual Function and Pure Virtual Function
10. C++ Exception Handling
11. C++ Templates
12. C++ Namespaces
13. C++ I/O and File

Books:

1. Teach Yourself C++ (Herbert Schildt)
2. The Complete Reference C++ (Herbert Schildt)







C++

Dr. Md. Humayun Kabir
CSE Department, BUET

History of C++

- Invented by Bjarne Stroustrup at Bell Lab in 1979
- Initially known as “C with Classes”
  - Classes and Basic Inheritance
- The name was changed to C++ in 1983
  - Function Overloading and Virtual Functions
- In 1985, Stroustrup's reference to the language entitled “*The C++ Programming Language”* was published
- Static Members and Inheritance from several Classes were included in 1989.

History of C++ (cont…..)

- In 1990, *The Annotated C++ Reference Manual* was released and Borland's Turbo C++ compiler was released as a commercial product.
- In 1998, the C++ standards committee published the first international standard for C++ ISO/IEC 14882:1998, known as C++98.
- The Standard Template Library was included in C++98.

History of C++ (cont…..)

- Problems reported with C98 standard were revised in 2003 and the changed language is known as C++03.
- In 2011, the new C++ standard, C++11, has been published.

History of C++ (cont…..)

- Some new features included in C++11 are
  - regular expression
  - a comprehensive randomization library
  - a new C++ time library
  - atomics support
  - a standard threading library (which both C and C++ were lacking)
  - a new for loop syntax providing functionality similar to foreach loops in certain other languages
  - auto keyword
  - new container classes
  - better support for unions and array-initialization lists
  - variadic templates.

C and C++

- C++ is the superset of C
- C++ includes everything of C and adds OOP support
- In addition C++ contains many improvements and features to make the language better than the original C.

New Style C++ Headers

- #include <iostream>
- #include <vector>
- #include <string>
- #include <cstring>
- Do not have **.h** extension and do not specify file name
- Specify standard identifiers that are mapped to files by the compiler
- C++ still support C-style header files, e.g.,
  - #include <stdio.h>
  - #include <string.h>

C++ Namespace

- A namespace is a declaration region
- It localizes the names of identifiers to avoid name collisions
- In C, the names of the library functions and other such items are not localized by any namespace, instead, they are implicitly placed into global namespace
- For this reason, C library functions with C-style headers can simply be used in programs without any special arrangement

C++ Namespace (cont…)

- In C++, the names of the library functions and other such items related to new-style headers are defined, i.e., localized, in the **std** namespace
- In order to use the library functions with new-style headers we need to bring std namespace into visibility

using namespace std;

being used

Namespace Name

C++ Console I/O

- In C++ I/O is performed using I/O operator instead of I/O functions
- The output operator is insertion operator, <<
- The input operator is extraction operator, >>
- In order to use either insertion (<<) or extraction (>>) operator programs must begin with the followings

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

C++ Console I/O (cont…)

- Insertion operator (<<) is associated with **cout**, a predefined stream linked to the console output (monitor)

```
cout<<"Hello!";
```

- Extraction operator (>>) is associated with **cin**, a predefined stream linked to the console input (keyboard)

```
int num;
cin>>num;
```

C++ Console I/O Code

```
#include <iostream>
using namespace std;
int main() {
      int num;
      cout<<“Enter an integer value: “;
      cin>>num;
      cout<<“Your entered number is “<<num<<“\n”;
      return 0;
}
```

C++ Console I/O Code Output

Enter an integer value: 100↵

Your entered number is 100

Concepts in Object-Oriented Programming

- Encapsulation
  - Information hiding
  - Objects contain their own copies of data and functions (algorithms)
- Polymorphism
  - A single name can have multiple meanings depending on its context
- Inheritance
  - Writing reusable code
  - Objects can inherit characteristics from other objects

C++ Features

- Supports data security
- Prevents accidents with data
- Helps code reuse.
- Lets you use operators the way you like.
- Allows multiple functions/operators with the same name.

Courtesy: Piyush Kumar, Florida State University

When to use C++

- Large projects
- System applications
- Graphics
- Data Structures
- Speed is an issue
- Changes in functionality required
- Need exceptions and error handling
- Want to speed up your scripts

Courtesy: Piyush Kumar, Florida State University

When not to use C++

- Small system programs (use C)
- Fast prototyping (use Visual Basic/Cold Fusion)
- Web-based applications (use PHP/Perl/Python)

Courtesy: Piyush Kumar, Florida State University

cout<<“Thank You”<<endl;

cout<<“**Have a Good Day**”<<endl;

Dr. Md. Toukir Ahmed, Assistant Professor, CSE, PUST