IT206 Data Structures Lab with OOP

Lecture 1

Rachit Chhaya

DA-IICT

March 28, 2022

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week
- Evaluation of lab n in lab $n+1$. Penalty for late submission
- Discussing with TA , classmates is allowed. Use of internet is allowed to check on syntax. **Directly copying code is strictly prohibited**.
- **Evaluation**:

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week
- Evaluation of lab n in lab $n+1$. Penalty for late submission
- Discussing with TA , classmates is allowed. Use of internet is allowed to check on syntax. **Directly copying code is strictly prohibited**.
- **Evaluation**:
  - Lab Exercises 20% (Code should be well documented).

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week
- Evaluation of lab n in lab $n+1$. Penalty for late submission
- Discussing with TA , classmates is allowed. Use of internet is allowed to check on syntax. **Directly copying code is strictly prohibited**.
- **Evaluation**:
  - Lab Exercises 20% (Code should be well documented).
  - Midsem 20%,

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week
- Evaluation of lab n in lab $n+1$. Penalty for late submission
- Discussing with TA , classmates is allowed. Use of internet is allowed to check on syntax. **Directly copying code is strictly prohibited**.
- **Evaluation**:
  - Lab Exercises 20% (Code should be well documented).
  - Midsem 20%,
  - Viva and/or surprise quizzes 30%.

Logistics

- Lecture every Monday or Wednesday (will be told beforehand) as per time table
- Batch wise 1 lab (2 hrs) every week
- Evaluation of lab n in lab $n+1$. Penalty for late submission
- Discussing with TA , classmates is allowed. Use of internet is allowed to check on syntax. **Directly copying code is strictly prohibited**.
- **Evaluation**:
  - Lab Exercises 20% (Code should be well documented).
  - Midsem 20%,
  - Viva and/or surprise quizzes 30%.
  - End Sem 30%.
- **Books**:Data Structures and Algorithms in C++ (Goodrich,Tamasssia et al.), Object oriented programming with C++ by Balagurusamy
- Feel free to use any material on internet only for **reference and with proper citation**

Content

- OOP concepts
- Realizing OOP concepts with C++

What we will not learn:

Content

- OOP concepts
- Realizing OOP concepts with C++

What we will not learn:
Theory of data structures (will be covered in IT205)

Questions?

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.
- Functions for different tasks

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.
- Functions for different tasks
- Global data

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.
- Functions for different tasks
- Global data

- **Object Oriented Programming**: Emphasis on data

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.
- Functions for different tasks
- Global data

- **Object Oriented Programming**: Emphasis on data
- Programs written in terms of objects.

POP vs OOP

- **Procedure oriented programming** - program is viewed as sequence of doing things.
- Functions for different tasks
- Global data

- **Object Oriented Programming**: Emphasis on data
- Programs written in terms of objects.
- Only functions attached to data can access the data.

OOP concepts

- **Class**: User defined data type with certain attributes and functions for data. For e.g: DAPerson.

  Can have attributes like name, id, age etc.

  Can have function like *display_details()*

OOP concepts

- **Class**: User defined data type with certain attributes and functions for data. For e.g: DAPerson.

  Can have attributes like name, id, age etc.

  Can have function like *display_details()*

- **Object**: Variable of the type Class. For e.g.: DAPerson ABC;

- **Data Abstraction**: Representing Essential features without giving too many details.

- **Data Abstraction**: Representing Essential features without giving too many details.
- Idea of abstraction gives rise to Abstract Data Types(ADT). For e.g: Class

- **Data Abstraction**: Representing Essential features without giving too many details.
- Idea of abstraction gives rise to Abstract Data Types(ADT). For e.g: Class

- **Encapsulation**:Wrapping up Data and its functions as single unit.

- **Data Abstraction**: Representing Essential features without giving too many details.
- Idea of abstraction gives rise to Abstract Data Types(ADT). For e.g: Class

- **Encapsulation**:Wrapping up Data and its functions as single unit.
- Information hiding

- **Inheritance**: Deriving one class from existing class.

- **Inheritance**: Deriving one class from existing class.
- E.g. DAFaculty, DAStudent are both derived from DAPerson.

- **Inheritance**: Deriving one class from existing class.
- E.g. DAFaculty, DAStudent are both derived from DAPerson.
- Allows reusability.

- **Inheritance**: Deriving one class from existing class.
- E.g. DAFaculty, DAStudent are both derived from DAPerson.
- Allows reusability.

- **Polymorphism**: Taking more than one form.

- **Inheritance**: Deriving one class from existing class.
- E.g. DAFaculty, DAStudent are both derived from DAPerson.
- Allows reusability.

- **Polymorphism**: Taking more than one form.
- E.g: Both DAFaculty and DAStudent can have function *show_grades()* but to a student it shows his own grade whereas to a faculty it may show grades of all students taking his/her course.

First C++ Code

```
#include <iostream>

int main() {
        int no1;
        std::cout << "Hello IT206 students!\n";
        std:: cin >> no1;
                std::cin.ignore();
        std::cout << "The number you entered is " << no1
        <<"\n";
        return 0;
    }
```

First C++ Code

```
#include <iostream>

int main() {
        int no1;
        std::cout << "Hello IT206 students!\n";
        std:: cin >> no1;
                std::cin.ignore();
        std::cout << "The number you entered is " << no1
        <<"\n";
        return 0;
    }
```

How To Compile

First C++ Code

```
#include <iostream>

int main() {
        int no1;
        std::cout << "Hello IT206 students!\n";
        std:: cin >> no1;
                std::cin.ignore();
        std::cout << "The number you entered is " << no1
        <<"\n";
        return 0;
    }
```

How To Compile

```
g++ sourcefile.cpp
To check output ./a.out
```

Example Code with Class (Taken from W3schools)

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

class Car {
public:
int speed(int maxSpeed);
};

int Car::speed(int maxSpeed) {
return maxSpeed;
}

int main() {
Car myObj;
cout << myObj.speed(200);
return 0;
}
```