



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

200
1822 - 2022

Encapsulation

Man Geni Addysg Uwch yng Nghymru
Where Higher Education Began in Wales

www.uwtsd.ac.uk



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David

200
1822 - 2022

Recap



<https://forms.office.com/e/rvbQYntQTk>

Man Geni Addysg Uwch yng Nghymru
Where Higher Education Began in Wales

www.uwtsd.ac.uk



Two fundamental elements in all C++ programs:

- **Program functions** – These are part of a program that performs actions.
- **Program data** – The data is the information of the program which gets affected by the program functions.

- Encapsulation is an Object Oriented Programming concept that binds together the data and functions .
- Keeping both safe from outside interference and misuse.
- Data encapsulation led to the important OOP concept of **data hiding**.

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

```
class Employee {
    private:
        // Private attribute
        int salary;

    public:
        // Setter
        void setSalary(int s) {
            salary = s;
        }
}
```

```
// Getter
int getSalary() {
    return salary;
}
};
```

```
int main() {
    Employee myObj;
    myObj.setSalary(50000);
    cout << myObj.getSalary();
    return 0;
}
```

encapsulation is achieved
through the use of access
specifiers

Why Encapsulation?

Data Protection: Encapsulation keeps data safe by making it private within the class. It cannot be directly accessed or modified from outside the class.

Controlled Access: Methods or functions are provided within the class to access and modify the data. These are the only approved ways to interact with the data, ensuring its integrity.

Data Validation and Security: Encapsulation allows you to implement checks and validations within the class methods, ensuring that only valid data is stored and maintained.

Abstraction: Encapsulation allows for hiding the complex details of implementation, providing a clear and simplified interface for using the class.

Design a Student class in C++ that encapsulates the necessary information for student records within a School Management System.

Create a Student class with the following specifications:

Private Data Members:

Student's name

Student's age

Student's unique identification number (ID)

Private member(s) for student grades or performance records

Public Member Functions:

A method to retrieve the student's age

A method to display the student's information

Ensure that access to sensitive information is restricted using access specifiers and that modifications to private data are performed through public member functions.

Solution:

```
#include <iostream>
#include <string>

class Student {
private:
    std::string name;
    int age;
    int studentID;
    // Private member for performance records
    float performanceRecord;
```

public:

// Constructor to initialize data members

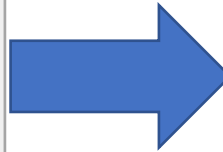
```
Student(std::string studentName, int studentAge, int id, float performance) : name(studentName),
    age(studentAge), studentID(id), performanceRecord(performance) {}
```

// Method to retrieve the student's age

```
int getAge() {  
    return age;  
}
```

// Method to display student's information

```
void displayInfo() {  
    std::cout << "Student Information:" << std::endl;  
    std::cout << "Name: " << name << std::endl;  
    std::cout << "Age: " << age << std::endl;  
    std::cout << "ID: " << studentID << std::endl;  
    std::cout << "Performance: " << performanceRecord;  
}  
};
```



```
int main()
```

```
{  
    Student student1("Alice", 18, 1001, 85.5);
```

```
    // Display initial information  
    student1.displayInfo();
```

```
    // Retrieve and display student's age  
    std::cout << "Student's age: " << student1.getAge();
```

```
    // Display updated information  
    student1.displayInfo();
```

```
    return 0;  
}
```

Next Session

- Assignment discussion
- Recap