**Al- Mustaqbal University**

**College of Sciences**

**Department of Cybersecurity**

**Object-Oriented Programming**

**(OOP)**

**(*Friend Functions and Friend Classes*)**

Second stage

Lecture 9/ Practical Part

2025-2024

 **Lectuer : Muntather AL-Mousawi**

**1- Friend Classes**

**ما هي دالة الصديق؟**

ولكن يتم منحها إذنًا للوصول إلى الأعضاء (class) دالة الصديق هي دالة ليست عضوًا في الصف

في هذا الصف. (protected) والمحمية (private)الخاصه

على الرغم من أنها ليست جزءًا من الصف، يمكنها الوصول إلى بياناته الخاصة وكأنها دالة عضو

**لماذا نستخدم دالة الصديق؟**

**حاجة دالة خارجية للوصول**: في بعض الأحيان، نحتاج إلى دالة خارجية للوصول إلى الأعضاء الخاصة في الصف لكننا لا نريد جعل هذه الأعضاء عامة

**تحسين المرونة**: يمكن أن تكون دالة الصديق مفيدة عندما تكون الدالة منطقياً مرتبطة بالصف لكنها لا تحتاج لأن تكون دالة عضو.

**طريقة كتابة دالة الصديق**

 ولكن يتم تعريفها خارج الصف friend يتم إعلان دالة الصديق داخل الصف باستخدام الكلمة المحجوزة

**(1)**

**Example**

#include <iostream>

using namespace std;

class Box {

**private:**

 int length;

**public:**

 Box(int len) : length(len) {} // Constructor to initialize length

 **friend void printLength(Box box);** // Friend function declaration

};

void printLength(Box box) { // Friend function definition

 cout << "Length of the box: " << box.length << endl;

} // Can access the private member of Box class

**int main() {**

 Box myBox(10);

 printLength(myBox); // Calling the friend function

 return 0;

}

**(2)**

**Example**

 #include <iostream>

using namespace std;

class Cylinder {

**private:**

 double radius;

 double height;

**public:**

 Cylinder(double r, double h) : radius(r), height(h) {}

 // Constructor to initialize radius and height

 friend double calculateVolume(Cylinder cyl); // Friend function declaration

};

double calculateVolume(Cylinder cyl) { // Friend function definition

 const double PI = 3.14159265; // Can access the private members of Cylinder class

 return PI \* cyl.radius \* cyl.radius \* cyl.height;

}

**int main() {**

 Cylinder myCylinder(3.0, 5.0);

 cout << "Volume of Cylinder: " << calculateVolume(myCylinder) << endl;

 // Calling the friend function

 return 0;

**(3)**

**Friend Functions**

 **؟ (Friend Class) ما هي فئة الصديق**

 ( Private ) هي فئة يتم منحها حق الوصول إلى الأعضاء الخاصة C++ فئة الصديق في

 لفئه اخرى عادةً تكون الأعضاء الخاصة للفئة غير قابلة للوصول من خارج الفئة (Protected)والمحميه ولكن فئة الصديق يمكنها الوصول إلى هذه الأعضاء. تُستخدم فئة الصديق عندما تكون فئتان مرتبطتين ارتباطًا وثيقًا وتحتاجان إلى مشاركة بعض التفاصيل الداخلية التي تكون مخفية عادةً

**لماذا نستخدم فئة الصديق؟**

**الفئات المرتبطة ارتباطًا وثيقًا**: في بعض الحالات، تكون هناك فئتان مرتبطتان ارتباطًا قويًا، وتحتاجان إلى التفاعل مع الأعضاء الخاصة لبعضهما البعض. فئة الصديق تسمح بهذا الوصول المباشر بينما لا تزال الفئات منفصلة

**العلاقات المعقدة**: عندما تتفاعل عدة فئات مع بعضها البعض بشكل معقد، يمكن أن يساعد استخدام فئات الصديق في تبسيط التحكم في الوصول بين هذه الفئات هذا يجنّب الحاجة إلى العديد من طرق الوصول لمجرد الوصول إلى الأعضاء الخاصة (getter/setter)

**صيغة فئة الصديق**

 داخل تعريف الفئة friend يتم الإعلان عن فئة صديقة باستخدام الكلمة المفتاحية

**(4)**

**Example**

#include <iostream>

using namespace std;

class Square; // Forward declaration of Square

class Rectangle {

**private:**

 int width, height;

**public:**

 Rectangle(int w, int h) : width(w), height(h) {}

 friend class Square; // Friend class declaration

};

class Square {

public:

 int calculateArea(Rectangle rect) {

 // Can access the private members of Rectangle

 return rect.width \* rect.height;

 }

};

**int main() {**

 Rectangle rect(4, 5);

 Square sq;

 cout << "Area of Rectangle: " << sq.calculateArea(rect) << endl;

 return 0;

**(5)**

**Example**

#include <iostream>

#include <string>

using namespace std;

class Book {

**private:**

 string title;

 string author;

 int pages;

**public:**

 // Constructor to initialize title and author

 Book(string t, string a, int p) : title(t), author(a), pages(p) {}

 friend class Library; // Declare Library as a friend class

};

class Library {

**public:**

 // Method to display book details, accessing private members of Book

 void displayBookDetails(const Book**&** book);

};

 // Friend class function definition

void Library::displayBookDetails(const Book& book) { // Friend class function definition

**(6)**

 // Access private members of Book

 cout << "Book Title: " << book.title << endl;

 cout << "Author: " << book.author << endl;

 cout << "No. of Pages: " << book.pages << endl;

}

**int main() {**

 Book myBook("Object oriented programming", "Abdulkadhem",350);

 Library myLibrary;

 myLibrary.displayBookDetails(myBook); // Using Library to display details of Book

 return 0;

}

**(7)**