Object Oriented Programming (OPP)

البرمجة كائنة التوجه (OPP)

Objects and Member Functions in Object Oriented Programming

الكائنات والدوال الأعضاء في البرمجة كائنية التوجه

Ali H. Hassan

Encapsulation

التغليف

Table of Contents

قائمة المحتويات

Introduction to Encapsulation

- In C++, OOPs encapsulation is implemented using classes and access specifiers that keeps the data, and the manipulating methods enclosed inside a single unit.
  - *في لغة C++ ، والبرمجة الكائنية التوجه (OPP) يطبق التغليف باستخدام فئات ومحددات وصول تُبقي البيانات وطرق المعالجة مُغلَّفة داخل وحدة واحدة.*
- The direct advantage of this packaging is that only the required components are visible to the user, and other information is hidden.
  - *الميزة المباشرة لهذا التغليف هي أن المكونات المطلوبة فقط هي المرئية للمستخدم، بينما تُخفى المعلومات الأخرى.*

What is Encapsulation?

- Encapsulation is one of the fundamental principles of object-oriented programming (OPP)
  - *التغليف هو واحد من المبادئ الرئيسية للبرمجة كائنية التوجه (OPP)*
- Provides data protection by restricting access
  - *يوفر حماية للبيانات من خلال تقييد الوصول إليها*
- Simplifies program maintenance
  - *يبسط صيانة البرنامج*

How to Encapsulation?

- The main idea of encapsulation is to hide data on the one hand and make it accessible on the other.
  - *فكرة التغليف الأساسية هي إخفاء البيانات من جهة و إتاحة التعامل معها من جهة أخرى*

- Variables must be made private
  - *المتغيرات يجب ان تكون خاصة*

- Find a way to access these variables from the outside
  - *ايجاد طريقة للوصول الى هذه المتغيرات من الخارج*

How to Encapsulation?

- Public functions should be prepared so that they can be accessed from anywhere.

  - *يجب تجهيز دوال نوعها عامة ليكون بالإمكان الوصول إليها من اي مكان*

- Properties should be kept private, while their accessor methods should be public

  - *اذا يجب جعل الخصائص من نوع خاص والدوال التي تستخدم في الوصول اليهم من النوع العام*

  - Setting a variable's value using set()
  - Getting a variable's value using get()

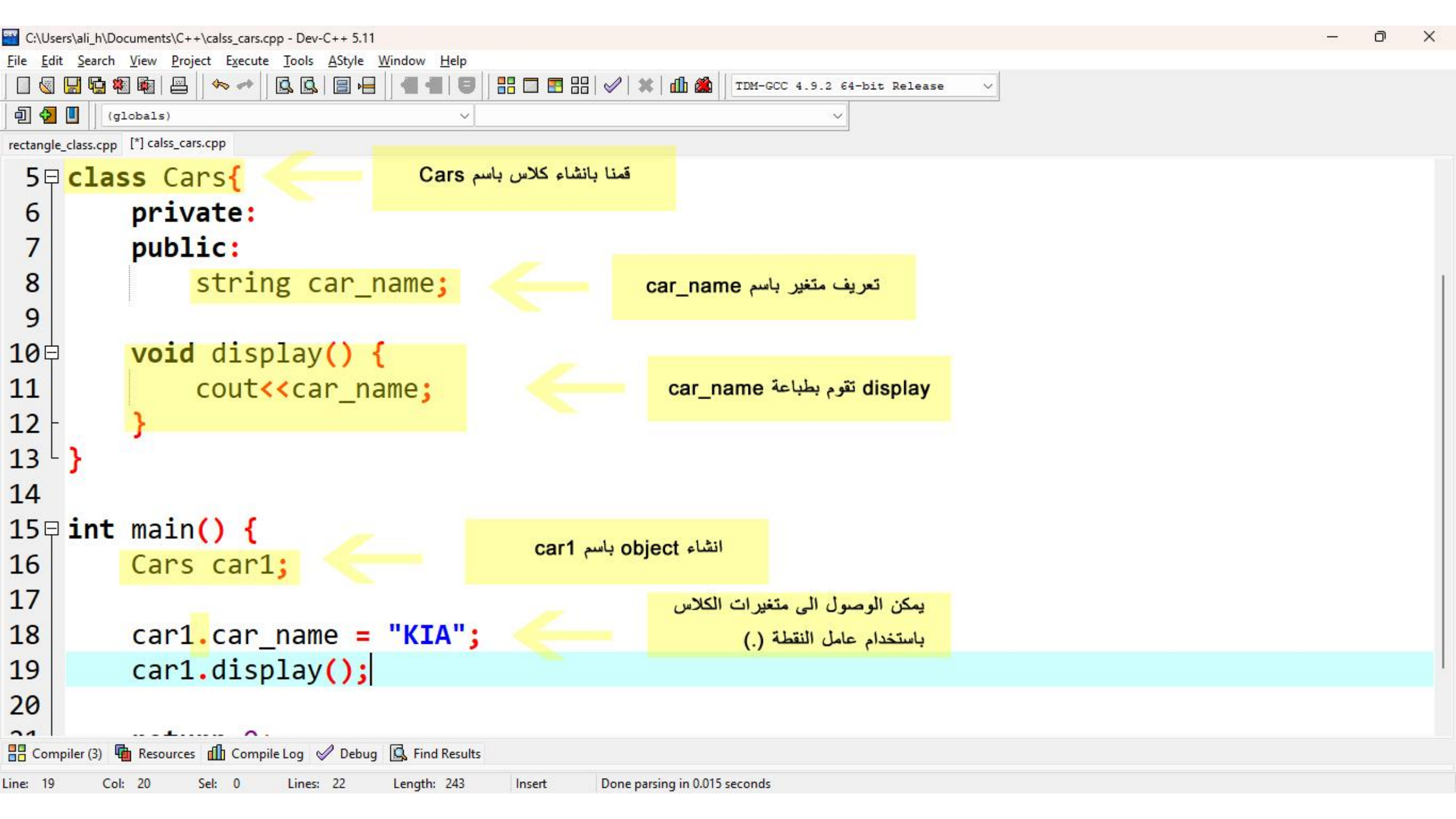
    - *اعطاء قيمة للمتغير باستخدام set()*
    - *الحصول على قيمة المتغير باستخدام get()*

Objects and Member Access

- When using objects, we access class members using the dot operator (.) for direct member access.
  - عندما نستخدم الكائنات , نستطيع الوصول الى أعضاء الكلاس, باستخدام عامل النقطة (.) للوصول المباشر لأعضاء الـclass

class_name object_name;

object_name.class_members;


C:\Users\ali_h\Documents\C++\calss_cars.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
rectangle_class.cpp
[*] calss_cars.cpp
class Cars{
private:
public:
string car_name;
void display() {
cout<<car_name;
}
}
int main() {
Cars car1;
car1.car_name = "KIA";
car1.display();
قمنا بانشاء كلاس باسم Cars
تعريف متغير باسم car_name
display تقوم بطباعة car_name
انشاء object باسم car1
يمكن الوصول الى متغيرات الكلاس
باستخدام عامل النقطة (.)
Compiler (3)
Resources
Compile Log
Debug
Find Results
Line: 19 Col: 20 Sel: 0 Lines: 22 Length: 243 Insert
Done parsing in 0.015 seconds

Defining Member Functions

Inside the class (Inline functions)

- Both member functions are defined inside the class itself, making them inline functions.
  - *يتم تعريف كلتا الدالتين الاعضاء داخل الكلاس نفسه، مما يجعلهما دالتين مضمنتين.*

```
class cars {
Public:

void setName(string n) {}

Int setPrice(int p) {}

};
```

C:\Users\ali_h\Documents\C++\Ali Haider\rectangle_class.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

[*] rectangle_class.cpp | [*] calss_cars.cpp | class_circle.cpp | rectangle_class_void.cpp | class_rectangle.cpp

```cpp
#include <iostream>

using namespace std;

class Rectangle {
    public:
        int width, length;

        void setDimensions(int i, int w) {
            length = i;
            width = w;
        }

        int calculateArea() {
            return length * width;
        }
};

int main() {
    Rectangle rect;

    rect.setDimensions(5, 10);

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

    return 0;
}
```

دالة setDimensions لحساب الابعاد
تم تعريفها داخل الكلاس Rectangle

دالة calculateArea تقوم بارجاع قيمة المساحة

Compiler (3) | Resources | Compile Log | Debug | Find Results

Line: 27 Col: 2 Sel: 0 Lines: 27 Length: 374 Insert Done parsing in 0 seconds

Defining Member Functions

Outside the class (Using scope resolution)

- Functions are defined outside the class using the scope resolution operator (::).
  - *يتم تعريف الدوال خارج الكلاس باستخدام عامل حل النطاق (::).*

- This is useful when you want to keep the class definition clean, especially for large functions.
  - *يُعد هذا مفيدًا عندما تريد الحفاظ على تعريف الكلاس نظيفًا، وخاصةً بالنسبة للدوال الكبيرة.*

```
class cars {
Public:

void setName(string n) {}

Int setPrice(int p) {}

};
```

```
void cars::setName(string n){
// code
}
```

C:\Users\ali_h\Documents\C++\Ali Haider\class_circle.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

[*] rectangle_class.cpp [*] calss_cars.cpp class_circle.cpp rectangle_class_void.cpp class_rectangle.cpp

```
4
5   class Circle {
6       private:
7           double radius;
8
9       public:
10
11          void setRadius(double r);
12
13          double calculateArea();
14
15  };
16
17  void Circle::setRadius(double r) {
18      radius = r;
19  }
20
21  double Circle::calculateArea() {
22      return 3.14 * radius * radius;
23  }
24
25  int main() {
26      Circle circle;
27
28      circle.setRadius(7);
29
30      cout<<"Area of the circle: "<<circle.calculateArea()<<endl;
31
32      return 0;
33  }
```

تم تعريف الدوال داخل الكلاس بدون تضمين عملها

تم الوصول الى الدوال من خارج الكلاس باستخدام ::
ومن ثم تم ذكر الية عمل كل دالة

Compiler (3) Resources Compile Log Debug Find Results

Line: 13 Col: 32 Sel: 0 Lines: 33 Length: 447 Insert Done parsing in 0 seconds

Object as Function Arguments

By Value

- The function receives a copy of the object, so changes made inside the function do not affect the original object
  - تتلقى الدالة نسخة من الكائن، وبالتالي فإن التغييرات التي يتم إجراؤها داخل الدالة لا تؤثر على الكائن الأصلي

```
class Cars {
Public:
string t;
void display() { cout<<t; }

};

void print (Cars c) {
c.t="Kia";
c.display();
}
```

```
int main() {
Cars car1;
Car1.t="BMW";
car1.display();
}
```

C:\Users\ali_h\Documents\C++\Ali Haider\printBook_class.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

[*] rectangle_class.cpp | [*] calss_cars.cpp | class_circle.cpp | rectangle_class_void.cpp | class_rectangle.cpp | printBook_class.cpp

```cpp
#include <iostream>

using namespace std;

class Book {
    public:
        string title;
        int pages;

        void setDetails(string t, int p) {
            title = t;
            pages = p;
        }

        void display() {
            cout<<"Title :"<<title<<", Pages: "<<pages<<endl;
        }
};

void printBook(Book b) {
    b.title="New Title";
    b.pages=398;
    b.display();
}

int main() {

    Book book1;

    book1.setDetails("C++ Programming", 300);
    printBook(book1);
    book1.display();


    return 0;
}
```

تحصل الدالة printBook على نسخة (b) من الـobject

وبالتالي فان التتغيرات التي يتم عملها داخل دالة printBook Book

لن تؤثر في محتوى الـobject الأصلي

output:

New Title

C++ Programming

Compiler (3) | Resources | Compile Log | Debug | Find Results

Line: 1 Col: 1 Sel: 0 Lines: 36 Length: 490 Insert Done parsing in 0.031 seconds

Object as Function Arguments

By Reference

- The Object is passed by reference. Any changes made inside the function affect the original object, as demonstrated by the change in the title.
  - *يتم تمرير الـobject بواسطة reference. أي تغييرات تُجرى داخل الدالة تؤثر على الـobject الأصلي.*

```
class Cars {
Public:
string t;
void display() { cout<<t; }

};

void print (Cars &c) {
c.t="Kia";
c.display();
}
```

```
int main() {
Cars car1;
Car1.t="BMW";
car1.display();
}
```

C:\Users\ali_h\Documents\C++\Ali Haider\modifyBook_class.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

[*] rectangle_class.cpp | [*] calss_cars.cpp | class_circle.cpp | rectangle_class_void.cpp | class_rectangle.cpp | printBook_class.cpp | modifyBook_class.cpp

```cpp
class Book {
    public:
        string title;
        int pages;

    void setDetails(string t, int p) {
        title = t;
        pages = p;
    }

    void display() {
        cout<<"Title: "<<title<<", Pages: "<<pages<<endl;
    }
};

void modifyBook(Book &b) {
    b.title = "Advanced C++";
    b.pages = 214;

    b.display();
}

int main() {

    Book book1;

    book1.setDetails("C++ Programming", 132);

    modifyBook(book1);

    book1.display();

    return 0;

    return 0;
}
```

تحصل الدالة modifyBook على نسخة أصلية (b) من الـobject

وبالتالي فان التتغيرات التي يتم عملها داخل دالة modifyBook Book

سوف تؤثر في محتوى الـobject الأصلي

output:

C++ Programming

C++ Programming

Compiler (3) | Resources | Compile Log | Debug | Find Results

Line: 1 Col: 1 Sel: 0 Lines: 40 Length: 513 Insert Done parsing in 0.015 seconds

Object as Return Type

- A function can also return an object. This is useful when we want to return a modified object or create new objects within a function.
  - يمكن للدالة أيضًا إرجاع كائن. وهذا مفيد عندما نريد إرجاع كائن معدل أو إنشاء كائنات جديدة داخل الدالة.

```
class Numbers {
Private:
Int x;
Public:

Numbers add(Complex c) {
Numbers result;
result.x = x + c.real;
return result;
}

};
```

C:\Users\ali_h\Documents\C++\Ali Haider\complex_class.cpp - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

[*] rectangle_class.cpp | [*] calss_cars.cpp | class_circle.cpp | rectangle_class_void.cpp | class_rectangle.cpp | printBook_class.cpp | modifyBook_class.cpp | [*] complex_class.cpp

```cpp
using namespace std;

class Complex {
    private:
        int real, imag;

    public:

        void setValues(int r, int i) {
            real = r;
            imag = i;
        }

        void display() {
            cout<<"Complex number: "<<real<<"+"<<imag<<"i"<<endl;
        }

        Complex add(Complex c) {

            Complex result;

            result.real = real + c.real;
            result.imag = imag + c.imag;

            return result;
        }
};

int main() {

    Complex c1, c2, sum;

    c1.setValues(3, 4);
    c2.setValues(5, 6);

    sum = c1.add(c2);

    sum.display();

    return 0;
}
```

عدد مركب

الاول حقيقي والاخر تخيلي

الدالة setValues تستعمل

لتعيين القيم للعددين

باختصار تقوم الدالة

باضافة قيم object

على object اخر

تقوم دالة add بارجاع object جديد يمثل مجموع عددين مركبين

العدد الاول هو real / imag

العدد الثاني هو c.real / c.imag

ومن ثم تقوم بارجاع object يمثل النتيجة result

قمنا بانشاء ثلاثة objects من الـclass اعلاه Complex

قمنا بادخال قيم لكل من الـobject الاول والثاني

الـobject الاخير sum هو يمثل قيم الـobject الاول c1 مجموعًا مع قيم الـobject الثاني c2

نقوم باستدعاء دالة display لطباعة القيم النهائية لكل من العددين من الـobject الاخير sum

Compiler | Resources | Compile Log | Debug | Find Results

Line: 38 Col: 14 Sel: 0 Lines: 43 Length: 577 Insert Could not find information for expression "c1.setValues"

Thank You