1- Write a program that defines a generic swap function to exchange the values of two variables. Demonstrate the usage of the generic swap function with integers and characters. Provide the program code and the output.

اكتب برنامجًا يحدد generic swap function لتبادل قيم متغيرين. توضيح استخدام وظيفة المبادلة العامة مع الأعداد الصحيحة والحروف. توفير رمز البرنامج والإخراج.

Output

```
After swapping integers: 20, 10
After swapping characters: B, A
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

template<typename T>
void swapValues(T &a, T &b) {
    T temp = a;
    a = b;
    b = temp;
}

int main() {
    int num1 = 10, num2 = 20;
    swapValues(num1, num2);
    cout << "After swapping integers: " << num1 << ", " << num2 << endl;

    char char1 = 'A', char2 = 'B';
    swapValues(char1, char2);
    cout << "After swapping characters: " << char1 << ", " << char2 << endl;

    return 0;
}
```

2- Write a program that utilizes a generic function to calculate the sum of two values. Demonstrate the generic sum function with integers, floats, and characters. Provide the program code and the output.

اكتب برنامجًا يستخدم generic function لحساب مجموع قيمتين. شرح generic sum function with integers, floats, and characters.
توفير رمز البرنامج والإخراج.

Output

```
Sum of integers: 8
Sum of floats: 7
Concatenation of characters: 
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

template<typename T>
T calculateSum(T a, T b) {
    return a + b;
}

int main() {
    cout << "Sum of integers: " << calculateSum(5, 3) << endl;
    cout << "Sum of floats: " << calculateSum(4.2, 2.8) << endl;
    cout << "Concatenation of characters: " << calculateSum('A', 'B') << endl;

    return 0;
}
```

3- Create a program that defines a generic function to find the maximum value in an array. Demonstrate the usage of the generic maximum function with an array of integers and an array of floats. Provide the program code and the output.

قم بإنشاء برنامج يقوم بتعريف generic function للعثور على القيمة القصوى في array . توضيح استخدام generic maximum function with an array of integers and an array of floats. توفير رمز البرنامج والإخراج.

Output

```
Maximum in integer array: 10
Maximum in float array: 6.7
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

template<typename T>
T findMax(T arr[], int size) {
    T maxVal = arr[0];
    for (int i = 1; i < size; ++i) {
        if (arr[i] > maxVal) {
            maxVal = arr[i];
        }
    }
    return maxVal;
}

int main() {
    int intArray[] = {3, 8, 2, 10, 5};
    cout << "Maximum in integer array: " << findMax(intArray, 5) << endl;

    float floatArray[] = {2.5, 4.8, 1.2, 6.7, 3.9};
    cout << "Maximum in float array: " << findMax(floatArray, 5) << endl;

    return 0;
}
```

4- Create a class named Employee with attributes name and salary. Include a method to give an annual raise of 10%.

قم بإنشاء class باسم "Employee" with attributes name and salary. قم بتضمين طريقة لمنح زيادة سنوية قدرها 10٪.

Output

```
Annual raise given. New salary: $55000
```

Solution

```cpp
// www.gammal.tech

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

class Employee {
public:
    string name;
    double salary;

    void giveRaise() {
        salary *= 1.10; // 10% raise
        cout << "Annual raise given. New salary: $" << salary << endl;
    }
};

int main() {
    Employee emp;
    emp.name = "John";
    emp.salary = 50000.0;

    emp.giveRaise();

    return 0;
}
```

5- Create a class named Time with attributes hours, minutes, and seconds. Include a method to display the time in HH:MM:SS format.

قم بإنشاء class تسمى "Time" with attributes hours, minutes, and seconds. قم بتضمين طريقة لعرض الوقت بتنسيق HH:MM:SS.

Output

```
Time: 14:30:45
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

class Time {
public:
    int hours;
    int minutes;
    int seconds;

    void displayTime() {
        cout << "Time: " << hours << ":" << minutes << ":" << seconds << endl;
    }
};

int main() {
    Time currentTime;
    currentTime.hours = 14;
    currentTime.minutes = 30;
    currentTime.seconds = 45;

    currentTime.displayTime();

    return 0;
}
```

6- Create a class named Square with an attribute sideLength. Include a method to calculate the perimeter of the square.

قم بإنشاء class باسم Square مع سمة SideLength. تضمين طريقة لحساب محيط المربع.

Output

```
Perimeter of the square: 24
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

class Square {
public:
    double sideLength;

    double calculatePerimeter() {
        return 4 * sideLength;
    }
};

int main() {
    Square square;
    square.sideLength = 6.0;

    cout << "Perimeter of the square: " << square.calculatePerimeter() << endl;

    return 0;
}
```

7- Create a class named Song with attributes title and artist. Include a method to play the song.

قم بإنشاء class باسم أغنية
with attributes title and artist. قم بتضمين طريقة لتشغيل الأغنية.

Output

```
Now playing: Shape of You by Ed Sheeran
```

Solution

```cpp
// www.gammal.tech

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

class Song {
public:
    string title;
    string artist;

    void play() {
        cout << "Now playing: " << title << " by " << artist << endl;
    }
};

int main() {
    Song mySong;
    mySong.title = "Shape of You";
    mySong.artist = "Ed Sheeran";

    mySong.play();

    return 0;
}
```

8- Create a base class BankAccount with attributes accountNumber and balance and a method displayBalance to display the account balance. Derive a class SavingsAccount from BankAccount with an additional attribute interestRate and a method calculateInterest to display the interest earned.

Output

```
Account Number: 12345, Balance: $1000
Interest earned: $50
```

Solution

```cpp
// www.gammal.tech

#include<iostream>
using namespace std;

class BankAccount {
public:
    int accountNumber;
    double balance;

    void displayBalance() {
        cout << "Account Number: " << accountNumber << ", Balance: $" << balance << endl;
    }
};

class SavingsAccount : public BankAccount {
public:
    double interestRate;

    void calculateInterest() {
        double interest = balance * interestRate / 100;
        cout << "Interest earned: $" << interest << endl;
    }
};

int main() {
    SavingsAccount mySavings;
    mySavings.accountNumber = 12345;
    mySavings.balance = 1000.0;
    mySavings.interestRate = 5.0;

    mySavings.displayBalance();
    mySavings.calculateInterest();

    return 0;
}
```

9- Create a program using inheritance. Define a base class Employee with attributes employeeId and salary and a method displayInfo to display the employee's information. Derive a class Manager from Employee with an additional attribute department and a method displayDepartment to display the manager's department. Get details from the user for both the employee and the manager.

Input

```
Enter the employee details:
Enter employee ID: 123
Enter salary: $50000
Enter the manager details:
Enter department: HR
```

Output

```
Employee ID: 123, Salary: $50000
Department: HR
```

Solution

```cpp
// www.gammal.tech

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

class Employee {
public:
    int employeeId;
    double salary;

    void displayInfo() {
        cout << "Employee ID: " << employeeId << ", Salary: $" << salary << endl;
    }
};

class Manager : public Employee {
public:
    string department;

    void displayDepartment() {
        cout << "Department: " << department << endl;
    }
};

int main() {
    Manager myManager;

    cout << "Enter the employee details:" << endl;
    cout << "Enter employee ID: ";
    cin >> myManager.employeeId;

    cout << "Enter salary: $";
    cin >> myManager.salary;

    cout << "Enter the manager details:" << endl;
    cout << "Enter department: ";
    cin.ignore(); // Consume the newline character in the buffer
    getline(cin, myManager.department);

    myManager.displayInfo();
    myManager.displayDepartment();

    return 0;
}
```

10- Create a program demonstrating inheritance. Define a base class Fruit with attributes name and color and a method displayInfo to display the fruit's information. Derive a class Apple from Fruit with an additional attribute taste and a method displayTaste to display the taste of the apple. Get details from the user for both the fruit and the apple.

Input

```
Enter the fruit details:
Enter fruit name: Apple
Enter color: Red
Enter the apple details:
Enter taste: Sweet
```

Output

```
Fruit: Apple, Color: Red
Taste: Sweet
```

Solution

```cpp
// www.gammal.tech

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

class Fruit {
public:
    string name;
    string color;

    void displayInfo() {
        cout << "Fruit: " << name << ", Color: " << color << endl;
    }
};

class Apple : public Fruit {
public:
    string taste;

    void displayTaste() {
        cout << "Taste: " << taste << endl;
    }
};

int main() {
    Apple myApple;

    cout << "Enter the fruit details:" << endl;
    cout << "Enter fruit name: ";
    getline(cin, myApple.name);

    cout << "Enter color: ";
    getline(cin, myApple.color);

    cout << "Enter the apple details:" << endl;
    cout << "Enter taste: ";
    getline(cin, myApple.taste);

    myApple.displayInfo();
    myApple.displayTaste();

    return 0;
}
```
