
1- Create a class SimpleConstructor with a default constructor that prints "Default Constructor Called" when an object is created.

Output

```
Default Constructor Called
```

Solution

```cpp
// www.gammal.tech
#include <iostream>
using namespace std;

class SimpleConstructor {
public:
    SimpleConstructor() {
        cout << "Default Constructor Called" << endl;
    }
};

int main() {
    SimpleConstructor obj;
    return 0;
}
```

2- Create a class ParameterizedConstructor with a parameterized constructor that takes an integer as input and prints its value.

Output

```
Parameterized Constructor Called with value: 42
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

class ParameterizedConstructor {
public:
    ParameterizedConstructor(int value) {
        cout << "Parameterized Constructor Called with value: " << value << endl;
    }
};

int main() {
    ParameterizedConstructor obj(42);
    return 0;
}
```

3- Create a class DefaultAndParameterized with both default and parameterized constructors. Display appropriate messages for each constructor

Output

```
Default Constructor Called
Parameterized Constructor Called with value: 24
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

class DefaultAndParameterized {
public:
    DefaultAndParameterized() {
        cout << "Default Constructor Called" << endl;
    }

    DefaultAndParameterized(int value) {
        cout << "Parameterized Constructor Called with value: " << value << endl;
    }
};

int main() {
    DefaultAndParameterized obj1;
    DefaultAndParameterized obj2(24);
    return 0;
}
```

4- Create a class MultipleConstructors with two constructors - one without parameters and another with two parameters. Display appropriate messages for each constructor.

Output

```
Constructor without parameters called
Constructor with parameters called: 10, 20
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

class MultipleConstructors {
public:
    MultipleConstructors() {
        cout << "Constructor without parameters called" << endl;
    }

    MultipleConstructors(int value1, int value2) {
        cout << "Constructor with parameters called: " << value1 << ", " << value2 << endl;
    }
};

int main() {
    MultipleConstructors obj1;
    MultipleConstructors obj2(10, 20);
    return 0;
}
```

5- Create a class UserInputConstructor with a parameterized constructor that takes input from the user and displays the entered value.

Input

```
Enter a value: 50
```

Output

```
Parameterized Constructor Called with value: 50
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

class UserInputConstructor {
public:
    UserInputConstructor() {
        int inputValue;
        cout << "Enter a value: ";
        cin >> inputValue;
        cout << "Parameterized Constructor Called with value: " << inputValue << endl;
    }
};

int main() {
    UserInputConstructor obj;
    return 0;
}
```

6- Create a class MultipleUserInput with a constructor that takes two inputs from the user and displays their sum.

Input

```
Enter two values: 5 6
```

Output

```
Sum of entered values: 11
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

class MultipleUserInput {
public:
    MultipleUserInput() {
        int value1, value2;
        cout << "Enter two values: ";
        cin >> value1 >> value2;
        cout << "Sum of entered values: " << (value1 + value2) << endl;
    }
};

int main() {
    MultipleUserInput obj;
    return 0;
}
```

7- Create a class UserData with a parameterized constructor that takes user's name and age as input and displays a greeting message.

Input

```
Enter your name: Amr
Enter your age: 20
```

Output

```
Hello, Amr! You are 20 years old.
```

Solution

```cpp
// www.gammal.tech

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

class UserData {
public:
    UserData() {
        string name;
        int age;
        cout << "Enter your name: ";
        getline(cin, name);
        cout << "Enter your age: ";
        cin >> age;
        cout << "Hello, " << name << "! You are " << age << " years old." << endl;
    }
};

int main() {
    UserData obj;
    return 0;
}
```

8- Create a class PersonDetails with a parameterized constructor that takes the name and age of a person from the user and displays the details.

Input

```
Enter your name: Ahmed
Enter your age: 15
```

Output

```
Person Details:
Name: Ahmed
Age: 15
```

Solution

```cpp
// www.gammal.tech

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

class PersonDetails {
public:
    PersonDetails() {
        string name;
        int age;

        cout << "Enter your name: ";
        getline(cin, name);
        cout << "Enter your age: ";
        cin >> age;

        cout << "Person Details:" << endl;
        cout << "Name: " << name << endl;
        cout << "Age: " << age << endl;
    }
};

int main() {
    PersonDetails obj;
    return 0;
}
```

9- Create a class named Gammal_Tech_Member to represent a member of Gammal Tech. The member has attributes such as name, cellphone number (cell), and the number of days (days) they have been a member. Implement a parameterized constructor in the class to initialize the name and cellphone number. The default values for the name and cellphone number should be set to "Gammal Tech New User" and an empty string, respectively. Also, set the initial number of days to 0. Implement a method print to print the member's name, cellphone number, and the number of days. In the main function, create an object of the class with a cellphone number, and print the member details.

Output

```
Name: Gammal Tech New User
Cellphone: +201033998844
Days of Membership: 0
```

Solution

```cpp
// www.gammal.tech

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

class Gammal_Tech_Member {
private:
    string name, cell;
    int days;

public:
    Gammal_Tech_Member(string c) {
        name = "Gammal Tech New User";
        cell = c;
        days = 0;
    }

    void print() {
        cout << endl << "Name: " << name << endl;
        cout << "Cellphone: " << cell << endl;
        cout << "Days of Membership: " << days << endl << endl;
    }
};

int main() {
    Gammal_Tech_Member x("+201033998844");
    x.print();
    return 0;
}
```

10- Create a class named Gammal_Tech_Member to represent a member of Gammal Tech. The member has attributes such as name, cellphone number (cell), and the number of days (days) they have been a member. Implement a parameterized constructor in the class to initialize the name and cellphone number. The default values for the name and cellphone number should be set to "Gammal Tech New User" and an empty string, respectively. Also, set the initial number of days to 0. Implement a method print to print the member's name, cellphone number, and the number of days. Additionally, add two methods: renewMembership to renew the membership by a specified number of days and updateCellphone to update the member's cellphone number. In the main function, create an object of the class with a cellphone number, renew the membership for 30 days, update the cellphone number, and print the updated member details.

Output

```
Original Details:

Name: Gammal Tech New User
Cellphone: +201033998844
Days of Membership: 0

Membership renewed for 30 days.
Cellphone number updated to: +19876543210
Updated Details:

Name: Gammal Tech New User
Cellphone: +19876543210
Days of Membership: 30
```

Solution

```cpp
// www.gammal.tech

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

class Gammal_Tech_Member {
private:
    string name, cell;
    int days;

public:
    Gammal_Tech_Member(string c) {
        name = "Gammal Tech New User";
        cell = c;
        days = 0;
    }

    void print() {
        cout << endl << "Name: " << name << endl;
        cout << "Cellphone: " << cell << endl;
        cout << "Days of Membership: " << days << endl << endl;
    }

    void renewMembership(int additionalDays) {
        days += additionalDays;
        cout << "Membership renewed for " << additionalDays << " days." << endl;
    }

    void updateCellphone(string newCell) {
        cell = newCell;
        cout << "Cellphone number updated to: " << newCell << endl;
    }
};

int main() {
    Gammal_Tech_Member x("+201033998844");

    // Original details
    cout << "Original Details:" << endl;
    x.print();

    // Renew Membership
    x.renewMembership(30);

    // Update Cellphone
    x.updateCellphone("+19876543210");

    // Updated details
    cout << "Updated Details:" << endl;
    x.print();

    return 0;
}
```