Task # 01

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

using namespace std;

class Course {
public:
    virtual void registerCourse() = 0;
};

class MathCourse : public Course {
public:
    void registerCourse() {
        cout << "Maths course has been registered!" << endl;
    }
};

class ProgrammingCourse : public Course {
public:
    void registerCourse() {
        cout << "Programming course has been registered!" << endl;
    }
};

int main() {
    Course *c1 = new MathCourse();
    Course *c2 = new ProgrammingCourse();

    c1->registerCourse();
    c2->registerCourse();
    return 0;
}
```

Task # 02

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

// Base class
class SmartDevice {
public:
    virtual void turnOn() = 0;
    virtual void turnOff() = 0;
    virtual ~SmartDevice() {}
};

// Derived class for Smart Light
class SmartLight : public SmartDevice {
public:
    void turnOn() override {
        cout << "Smart Light is turned ON.\n";
    }

    void turnOff() override {
        cout << "Smart Light is turned OFF.\n";
    }
};

// Derived class for Smart Fan
class SmartFan : public SmartDevice {
public:
    void turnOn() override {
        cout << "Smart Fan is spinning now.\n";
    }

    void turnOff() override {
        cout << "Smart Fan is stopped.\n";
    }
};

int main() {
    SmartDevice* device;

    SmartLight light;
    SmartFan fan;

    device = &light;
    device->turnOn();
    device->turnOff();

    cout << endl;
```

```
    device = &fan;
    device->turnOn();
    device->turnOff();

    return 0;
}
```

Task # 03

```cpp
#include <iostream>
#include <stdexcept> // runtime error
using namespace std;

// Function to validate marks
void validateMarks(int marks) {
    if (marks < 0 || marks > 100) {
        throw runtime_error("Invalid input: Marks should be between 0 and
100");
    }
}

int main() {
    int marks;

    cout << "Enter marks for the subject (0 to 100): ";
    cin >> marks;

    try {
        validateMarks(marks);
        cout << "Marks entered: " << marks << endl;
    }
    catch (const runtime_error& e) {
        cout << "Error: " << e.what() << endl;
    }

    return 0;
}
```

Task # 04

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

class FoodOrder {
public:
    void placeOrder() {
        cout << "Placing order...\n";
        checkStock();
        cookFood();
        generateInvoice();
        cout << "Order placed successfully!\n";
    }

private:
    void checkStock() {
        cout << "Checking stock...\n";
    }

    void cookFood() {
        cout << "Cooking food...\n";
    }

    void generateInvoice() {
        cout << "Generating invoice...\n";
    }
};

int main() {
    FoodOrder order;
    order.placeOrder();
    return 0;
}
```

Task # 05

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

void checkEligibility(int age) {
    if (age < 18) {
        throw runtime_error("Age is below 18. Not eligible for license.");
    }
    cout << "Eligible for driving license!" << endl;
}

int main() {
    int age;
    cout << "Enter your age: ";
    cin >> age;

    try {
        checkEligibility(age);
    }
    catch (const runtime_error& e) {
        cout << "Error: " << e.what() << endl;
    }

    return 0;
}
```