Programming Fundamentals

Lab Manual - Week 06

Introduction

Welcome to your favorite programming Lab. In this lab manual, we shall work together to learn and implement new programming concepts.

Skills to be learned:

- Write complex conditions with logical and comparison operators.
- Draw decision trees from complex real-world problems.
- Convert real-world problems with complex conditions (decision trees) into the code.

Let's do some coding.

Skill: Write complex conditions with logical and comparison operators.

Introduction

We have learned the multiple IF Block with a single boolean expression. Now, it's time to learn how to solve the same boolean expressions with IF-Else block if the conditions are contradicting.
Consider the following task for better understanding.

Task (WP): Write a C++ function that takes marks as input and then returns “You are Passed with XXX marks” if the marks are greater than 50 and returns “You are Failed with XXX marks” if the marks are less than 50 or equal to 50.

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

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

string passOrFail(int marks);

int main() {
    int score;

    cout << "Enter your score: ";
    cin >> score;

    cout << passOrFail(score);

    return 0;
}

string passOrFail(int marks)
{
    string result;
    if(marks > 50)
    {
        result = "You are Passed with " + to_string(marks) + " marks";
    }
    if(marks <= 50)
    {
        result = "You are Failed with " + to_string(marks) + " marks";
    }
    return result;
}
```

Following are the different comparison operators that are used in boolean expressions.

Operator	Description	Example
>	Greater than	`if (a > b)`
<	Less than	`if (a < b)`
>=	Greater than or equal to	`if (a >= b)`
<=	Less than or equal to	`if (a <= b)`

Skill: Write complex conditions with logical and comparison operators

==	if is equal to	`if (a == b)`

Task (WP): Now Consider the same task but with IF-Else block.

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

string passOrFail(int marks);

int main() {
    int score;

    cout << "Enter your score: ";
    cin >> score;

    cout << passOrFail(score);

    return 0;
}

string passOrFail(int marks)
{
    string result;
    if(marks > 50)
    {
        result = "You are Passed with " + to_string(marks) + " marks";
    }
    else
    {
        result = "You are Failed with " + to_string(marks) + " marks";
    }
    return result;
}
```

Only one of the blocks will be executed depending on the IF condition.

- If the IF condition is true then the IF block will be executed
- but if the condition in IF block is not true then the Else block will be executed.

Do the tasks using IF-ELSE Block

Task 01(OP): (greaterNumber)

Create a function that takes 2 integer numbers as input and returns true if the first number is greater than the second number otherwise it returns false.

greaterNumber(1, 4) → false

Skill: Write complex conditions with logical and comparison operators

greaterNumber(6, 5) ➞ true

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task1.exe
Enter the first number: 6
Enter the second number: 4
1
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task1.exe
Enter the first number: 1
Enter the second number: 9875
0
```

Task 02(OP): (parityAnalysis)
Create a function that takes a number as input and returns true if the sum of its digits has the same parity as the entire number. Otherwise, return false. **(Always take 3 digit number as input)**

Examples

parityAnalysis(243) ➞ true
// 243 is odd and so is 9 (2 + 4 + 3)

parityAnalysis(102) ➞ false
// 12 is even but 3 is odd (1 + 0 + 2)

parityAnalysis(300) ➞ false
// 300 is even but 3 is odd (3)

Notes

- Parity is whether a number is even or odd. If the sum of the digits is even and the number itself is even, return true. The same goes if the number is odd and so is the sum of its digits.

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task2.exe
Enter a 3-digit number: 123
0
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task2.exe
Enter a 3-digit number: 432
0
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task2.exe
Enter a 3-digit number: 222
1
```

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

Multiple IF Block:

A multiple if block allows a program to evaluate and handle different scenarios by checking multiple conditions and executing different code blocks accordingly. It is used to create branching logic in a program, where different paths are taken based on the outcomes of the conditions.

Task03(OP): (Perimeter)

Write a function that takes a number and returns the perimeter of following shapes.

Shape	Character	Perimeter Formula
Square	s	Perimeter of a square: 4 * side.
Circle	c	Perimeter of a circle: 6.28 * radius.
Equilateral Triangle	t	Perimeter of a triangle: 3 * side.
Regular Hexagon	h	Perimeter of a Hexagon: 6 * side.

The input will be in the form (letter l, number num) where the letter will be either "s" for *square*, or "c" for *circle*, or "t" for *triangle*, or "h" for *hexagon* and the number will be the side of the square or the radius of the circle or side of the triangle or side of the hexagon.

Examples

perimeter("s", 7) → 28
perimeter("c", 4) → 25.12
perimeter("c", 9) → 56.52

```
Enter the shape (s for square, c for circle, t for triangle, h for hexagon): t
Enter the value: 3.4
The perimeter is: 10.2

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task3.exe
Enter the shape (s for square, c for circle, t for triangle, h for hexagon): h
Enter the value: 4.6
The perimeter is: 27.6
```

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

Nested IF Block

An IF Condition inside an IF condition is called a Nested IF Condition.

Consider the task below

Task (WP): Write a c++ program that helps the user decide if he/she can buy a certain dress or not. The conditions are that the dress must cost less than 1500 **and** it must be from MTJ brand.

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

string canBuyDress(float dressCost, string brand);

int main()
{
    float dressCost;
    string brand;

    // Prompt the user for input.
    cout << "Enter the dress cost: $";
    cin >> dressCost;

    cout << "Enter the dress brand: ";
    cin >> brand;

    // Call the function to check if the user can buy the dress.
    cout << canBuyDress(dressCost, brand);

    return 0;
}

// Function to determine if the user can buy a dress.
string canBuyDress(float dressCost, string brand)
{
    if (dressCost < 1500.0)
    {
        if (brand == "MTJ")
        {
            return "Congratulations! You can buy the dress.";
        }
    }
    return "Sorry, the dress doesn't meet the criteria for purchase.";
}
```

Notice that we have used the **if condition inside an if condition** in the boolean expression. This is referred to as the **Nested IF Blocks.** We can check many conditions using the Nested IF Blocks.

We can write the solution of the same problem using && logical operator as well.

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

Task 01(WP): Write a C++ program that helps the user decide if he/she can buy a certain dress or not. The conditions are that the dress must cost less than 1500 **and** it must be from MTJ brand.

IF it is an MTJ kurta and **IF** the cost of the dress is less than 1500 **Then** the user should buy that dress.

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

string canBuyDress(float dressCost, string brand);

int main()
{
    float dressCost;
    string brand;

    // Prompt the user for input.
    cout << "Enter the dress cost: $";
    cin >> dressCost;

    cout << "Enter the dress brand: ";
    cin >> brand;

    // Call the function to check if the user can buy the dress.
    cout << canBuyDress(dressCost, brand);

    return 0;
}

// Function to determine if the user can buy a dress.
string canBuyDress(float dressCost, string brand)
{
    if (dressCost < 1500.0 && brand == "MTJ")
    {
        return "Congratulations! You can buy the dress.";
    }
    return "Sorry, the dress doesn't meet the criteria for purchase.";
}
```

Notice that we have used **&& operator** in the boolean expression. This is referred to as the **Logical AND operator.** We can check multiple conditions using the && operator, however, now the body of the IF Block will be executed **only if all the conditions with && operator are true**.

Skill: Write complex conditions with logical and comparison operators

Following are the logical operators that are often used in boolean expressions.

Operator	Description	Example
Not **!**	It inverts the results of the boolean expression	`if (a != b)`
AND **&&**	It returns True only if all conditions are True	`if (a > b && a > c)`
OR **||**	It returns True if any condition is True	`if (a > b || a > c)`

Task 04(CL): (findGreatest)
Write a program that inputs three numbers from the user and prints the greater number. Use logical operators. Function name should be **findGreatest** and it should take the three numbers as input and returns the greatest number.

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task4.exe
Enter the first number: 3
Enter the second number: 4
Enter the third number: 2
The greatest number among 3, 4, and 2 is: 4

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task4.exe
Enter the first number: 60
Enter the second number: 43
Enter the third number: 70
The greatest number among 60, 43, and 70 is: 70
```

Conclusion

We can use the **comparison operators** with the **logical operators** to solve **complex boolean expressions**. In addition, there are various **conditional statements** that can be used to evaluate different conditions.

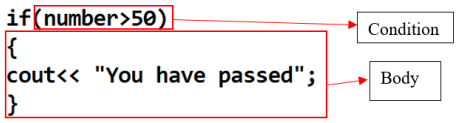
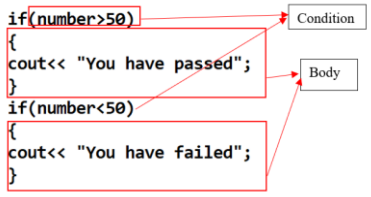
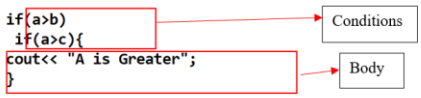
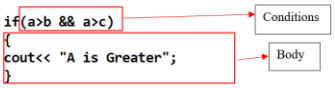
The following table highlights the difference between the different **conditional statements**.

Statement	Description

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

IF	 **Executes the body only if the condition is True.**
Multiple IF	 **Executes only those IFs that are True.**
IF ELSE	 **Executes IF otherwise ELSE**
Nested IF	 **The body is executed only if nested conditions are True.**
Logical operators	 **The body is executed depending on the evaluation of the Logical Operator used.**
Congratulations Students! You have just added another very important programming skill in your skillset. Keep up the learning pace. <3	

Skill: Write complex conditions with logical and comparison operators

Programming Fundamentals

Lab Manual - Week 06

Skill: Draw decision trees from complex real-world problems

Introduction

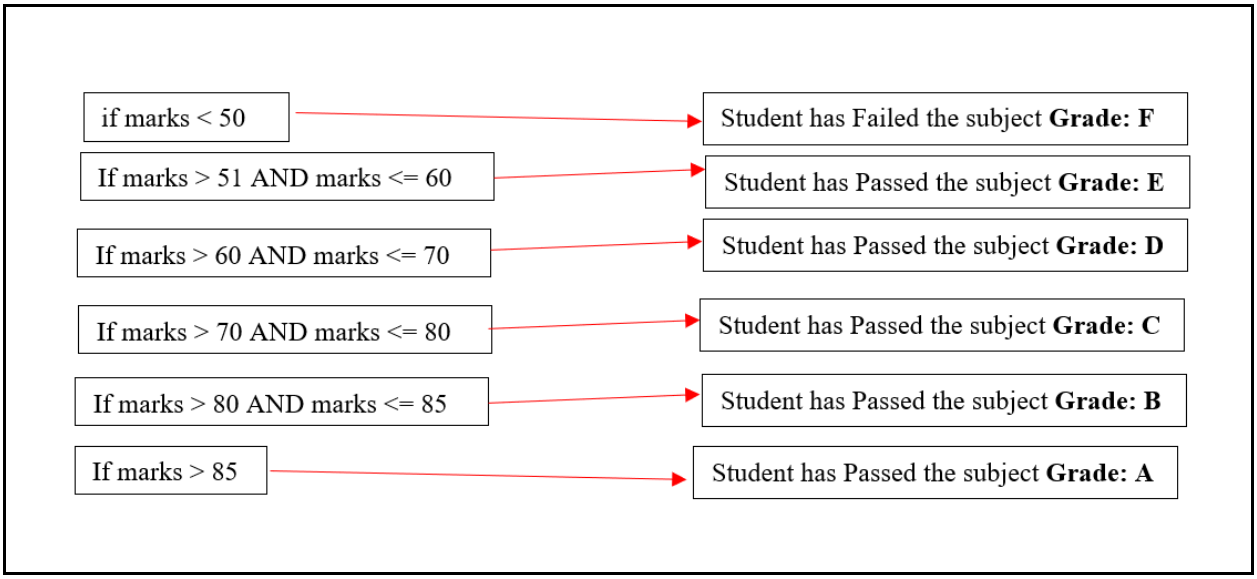
We can use the IF-ELSE Block to check **multiple conditions simultaneously**.

Consider the following task for a better understanding.

Task (WP): Write a C++ program that inputs the marks from the user and assigns the grade according to the following criteria.

Marks	**Grade**
<50	F
50-60	E
61-70	D
71-80	C
81-85	B
>85	A

Students, Based on the given information we can draw a decision tree such as follows.



In such a representation, we define what are the different conditions and what will happen if these conditions are satisfied respectively. However, the key takeaway is that **only one**

Skill: Draw decision trees from complex real-world problems

of all these **conditional blocks** will be true and the rest will be false and not executed by the compiler. For example, if the user has entered 45, the first condition will be true and it should print **Grade: F** and the rest of the conditions will be false.

Congratulations! You have just learned how to draw decision trees for complex real-world problems.

Programming Fundamentals

Lab Manual - Week 06

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Let's code the above-defined decision tree using **multiple IFs and Logical Operators.**

```cpp
#include <iostream>
using namespace std;
// Function to calculate and assign the grade based on marks.
char assignGrade(int marks)
{
    char grade;
    if (marks < 50) {
        grade = 'F';
    }
    if (marks >= 50 && marks <= 60) {
        grade = 'E';
    }
    if (marks >= 61 && marks <= 70) {
        grade = 'D';
    }
    if (marks >= 71 && marks <= 80) {
        grade = 'C';
    }
    if (marks >= 81 && marks <= 85) {
        grade = 'B';
    }
    if(marks > 85){
        grade = 'A';
    }
    return grade;
}

int main() {
    int marks;
    // Prompt the user for input.
    cout << "Enter the marks: ";
    cin >> marks;
    // Call the function to assign the grade.
    char grade = assignGrade(marks);
    // Display the assigned grade.
    cout << "Grade: " << grade;
}
```

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Programming Fundamentals

Lab Manual - Week 06

Task 05(OP): (Store V1)

A Store has announced to give a 10% discount on the total purchase amount every Sunday of October, and a 5% discount every other Sunday.

Write a Function that takes Day, Month, and total amount as input and returns the payable amount after the discount.

Test Cases:

Input	Output
discount("Sunday", "October", 4000);	3600
discount("Tuesday", "October", 4000);	4000
discount("Sunday", "November", 4000);	3800

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task5.exe
Enter Purchase Day: Sunday
Enter Purchase Month: November
Enter Purchase Amount: 4000
Payable Amount after discount: 3800
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task5.exe
Enter Purchase Day: Sunday
Enter Purchase Month: October
Enter Purchase Amount: 4000.457
Payable Amount after discount: 3600.41
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task5.exe
Enter Purchase Day: Monday
Enter Purchase Month: October
Enter Purchase Amount: 384.62
Payable Amount after discount: 384.62
```

Task 06 (OP): (Store V2)

A Store has announced to give a 10% discount on the total purchase amount on every Sunday or Month is October.

Write a Function that takes Day, Month, and total amount as input and returns the payable amount after the discount.

Test Cases:

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Input	Output
discount("Sunday", "October", 4000);	3600

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task6.exe
Enter Purchase Day: Sunday
Enter Purchase Month: December
Enter Purchase Amount: 2000.5
Payable Amount after discount: 1800.45
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task6.exe
Enter Purchase Day: Monday
Enter Purchase Month: October
Enter Purchase Amount: 600.54
Payable Amount after discount: 540.486
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task6.exe
Enter Purchase Day: Tuesday
Enter Purchase Month: July
Enter Purchase Amount: 3000
Payable Amount after discount: 3000
```

Task 07 (OP): (Store V3)

A Store has announced to give a 10% discount on the total purchase amount every Sunday of Month October, or March, or August.

Write a Function that takes Day, Month, and total amount as input and returns the payable amount after the discount.

Test Cases:

Input	Output
discount("Sunday", "August", 4000);	3600
discount("Tuesday", "October", 4000);	4000

Skill: Convert real-world problems with complex conditions (decision trees) into the code

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task7.exe
Enter Purchase Day: Sunday
Enter Purchase Month: February
Enter Purchase Amount: 1000.5
Payable Amount after discount: 1000.5
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task7.exe
Enter Purchase Day: Sunday
Enter Purchase Month: August
Enter Purchase Amount: 200.54
Payable Amount after discount: 180.486
```

Task 08 (OP): (Store V4)

A Store has announced to give a 10% discount on the total purchase amount every Sunday of Month October, or March, or August and a 5% discount on the total purchase amount every Monday of November or December.

Write a Function that takes Day, Month, and total amount as input and returns the payable amount after the discount.

Test Cases:

Input	Output
discount("Sunday", "August", 4000);	3600
discount("Tuesday", "October", 4000);	4000
discount("Monday", "November", 4000);	3800

Skill: Convert real-world problems with complex conditions (decision trees) into the code

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task8.exe
Enter Purchase Day: Monday
Enter Purchase Month: November
Enter Purchase Amount: 300.54
Payable Amount after discount: 285.513
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task8.exe
Enter Purchase Day: Monday
Enter Purchase Month: January
Enter Purchase Amount: 300.54
Payable Amount after discount: 300.54
```

Congratulations Students! You have successfully practiced and learned conditional statements. Now, let's do some problem-solving.

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Programming Fundamentals

Lab Manual - Week 06

Task 9(CL): (checkTitle)

Write a C++ function for the following question.

Depending on age (decimal number and gender (m / f), print a personal title:

- "Mr." – a man (gender "m") – 16 or more years old.
- "Master" – a boy (gender "m") under 16 years.
- "Ms." – a woman (gender "f") – 16 or more years old.
- "Miss" – a girl (gender "f") under 16 years.

Input	Output	Input	Output
12 f	Miss	17 m	Mr.

Input	Output	Input	Output
25 f	Ms.	13.5 m	Master

Below is a sample function prototype that can be used to solve this problem.

string checkTitle(int, char);

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task9.exe
Enter your age: 18
Enter your gender (m/f): m
Your personal title is: Mr.
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task9.exe
Enter your age: 18
Enter your gender (m/f): f
Your personal title is: Ms.
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task9.exe
Enter your age: 25
Enter your gender (m/f): f
Your personal title is: Ms.
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task9.exe
Enter your age: 12
Enter your gender (m/f): f
Your personal title is: Miss
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task9.exe
Enter your age: 12
Enter your gender (m/f): m
Your personal title is: Master
```

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Programming Fundamentals

Lab Manual - Week 06

Task 10(OP): (areSameNumber)

Write a program that inputs 3 numbers and prints whether they are the same (yes/no).

Input	Output
5 5 5	yes
5 4 5	no
1 2 3	no

Below is a sample function prototype that can be used to solve this problem.

bool areSameNumber(int, int, int);

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task10.exe
Enter the first number: 70
Enter the second number: 70
Enter the third number: 70
1
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task10.exe
Enter the first number: 20
Enter the second number: 32
Enter the third number: 10
0
```

Task 11(CP): (checkSpeed)

Write a program that inputs the speed (decimal number) and prints speed information. For speed up to 10 (inclusive), print "slow". For speed over 10 and up to 50, print "average". For speed over 50 and up to 150, print "fast". For speeds over 150 and up to 1000, print "ultra-fast". For higher speed, print "extremely fast".

Skill: Convert real-world problems with complex conditions (decision trees) into the code

Input	Output
8	slow
49.5	average
126	fast
160	ultra fast
3500	extremely fast

Below is a sample function prototype that can be used to solve this problem.

string checkSpeed(float);

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task11.exe
Enter the speed: 10
slow
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task11.exe
Enter the speed: 150
fast
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task11.exe
Enter the speed: 151
ultra-fast
```

Task 12(CP): (totalIncome)

In a cinema hall the chairs are ordered in a rectangle shape in r rows and c columns. There are three types of screenings with tickets of different prices:

- Premiere – a premiere screening, with a price of 12.00 EUR.
- Normal – a standard screening, with a price of 7.50 EUR.
- Discount – a screening for children and students at a reduced price – 5.00 EUR.

Write a function that enters a type of screening (string), the number of rows, and the number of columns in the hall (integer numbers) and returns the total income from tickets from a full hall.

Input	Output	Input	Output
Premiere 10 12	1440.00	Normal 21 13	2047.50

Below is a sample function prototype that can be used to solve this problem.

float totalIncome(string, int, int);

Skill: Convert real-world problems with complex conditions (decision trees) into the code

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task12.exe
Enter the screening type (Premiere/Normal/Discount): Premiere
Enter the number of rows: 10
Enter the number of columns: 12
1440
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task12.exe
Enter the screening type (Premiere/Normal/Discount): Normal
Enter the number of rows: 21
Enter the number of columns: 13
2047.5
```

Task 13(CP): (lowestPrice)

A student has to travel n kilometers. He can choose between three types of transportation:

- Taxi. Starting fee: 0.70 EUR. Day rate: 0.79 EUR/km. Night rate: 0.90 EUR/km.
- Bus. Day / Night rate: 0.09 EUR/km. It can be used for distances of a minimum of 20 km.
- Train. Day / Night rate: 0.06 EUR/km. It can be used for distances of a minimum of 100 km.

Write a program that reads the number of kilometers n and period of the day (day or night) and calculates the price for the cheapest transport.

Input data:
number n – number of kilometers – an integer in the range of [1 … 5000].
the word “day” or “night” – traveling during the day or during the night.

Output data:
Print the lowest price for the given number of kilometers on the console.

Input	Output	Input	Output
5 day	4.65	7 night	7

Input	Output	Input	Output
25 day	2.25	180 night	10.8

Below is a sample function prototype that can be used to solve this problem.

float lowestPrice(int, string);

Skill: Convert real-world problems with complex conditions (decision trees) into the code

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task13.exe
Enter the number of kilometers: 5
Enter the period of the day (day/night): day
Lowest price for 5 kilometers: 4.65 EUR

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task13.exe
Enter the number of kilometers: 7
Enter the period of the day (day/night): night
Lowest price for 7 kilometers: 7 EUR

G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task13.exe
Enter the number of kilometers: 25
Enter the period of the day (day/night): day
Lowest price for 25 kilometers: 2.25 EUR
```

Task 14(CP): (`calculateCost`)

A group of football fans decided to buy tickets for the Football World Cup 2022. The tickets are sold in Qatari Rial (QR) in two price categories:

- VIP – 499.99 QR (Qatari Rial)
- Normal – 249.99 Qatari Rial (Qatari Rial)

The football fans have a shared budget, and the number of people in the group determines what percentage of the budget will be spent on transportation:

- 1 to 4 – 75% of the budget
- 5 to 9 – 60% of the budget
- 10 to 24 – 50% of the budget
- 25 to 49 – 40% of the budget
- 50 or more – 25% of the budget

Write a program that calculates whether the money left in the budget will be enough for the football fans to buy tickets in the selected category, as well as how much money they will have left or be insufficient.

Input Data

The input data is read from the console and contains exactly 3 lines:

- The first line contains the budget – real number within the range [1 000.00 … 1 000 000.00].
- The second line contains the category – "VIP" or "Normal".

Skill: Convert real-world problems with complex conditions (decision trees) into the code

- The third line contains the number of people in the group – an integer within the range [1 … 200].

Output Data

Print the following on the console as one line:

- If the budget is sufficient:
  - "Yes! You have {N} leva left." – where N is the amount of remaining money for the group.
- If the budget is NOT sufficient:
  - "Not enough money! You need {M} leva." – where M is the amount that is insufficient.

The amounts must be formatted up to the second digit after the decimal point.

Sample output:

Input	Output
1000 Normal 1	Yes! You have 0.01 QR left.
30000 VIP 49	Not enough money! You need 6499.51 QR.

Function prototype should be:

```
string calculateCost(float budget, string category, int numPeople);
```

Skill: Convert real-world problems with complex conditions (decision trees) into the code

```
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task14.exe
Enter the budget: 1000
Enter the category (VIP/Normal): Normal
Enter the number of people in the group: 1
Yes! You have 0.010000 leva left.
G:\Semesters\Programming Fundamentals (Fall 2023)\Week 6\Lab Tasks>Task14.exe
Enter the budget: 30000
Enter the category (VIP/Normal): VIP
Enter the number of people in the group: 49
Not enough money! You need 6499.510000 leva.
```

Good Luck and Best Wishes !!

Happy Coding ahead :)

Skill: Convert real-world problems with complex conditions (decision trees) into the code