
# CS110: LAB 03

## Conditional Statements-Decision Making

**Muhammad Mujtaba**

**CMD ID: 540040**

mmujtaba.bese25seecs@seecs.edu.pk

**Class:** BESE 16B

**Batch:** 2k25

# Task 1 [CLO 1]:

## CODE

```cpp
#include <iostream>

// utility; mainly used to improve readability
inline bool isDivisible(int input, int checkNum)
{
    return input % checkNum == 0;
}

int main()
{
    int input_num;

    std::cout << "Enter a Number to check divisibility of: ";
    std::cin >> input_num;
    std::cout << std::endl;

    if (isDivisible(input_num, 2))
    {
        std::cout << "You entered an EVEN number (" << input_num << ")\n";
        if (isDivisible(input_num, 4))
        {
            std::cout << input_num << " is divisible by 4\n";
        }
        if (isDivisible(input_num, 6))
        {
            std::cout << input_num << " is divisible by 6\n";
        }
    }
    else
    {
        std::cout << "You entered an ODD number (" << input_num << ")\n";
        if (isDivisible(input_num, 3))
        {
            std::cout << input_num << " is divisible by 3\n";
        }
        if (isDivisible(input_num, 5))
        {
            std::cout << input_num << " is divisible by 5\n";
        }
    }


    // ignore is used to clear previous inputs from user
    std::cin.ignore();
    std::cin.get();
    return 0;
}
```

## OUTPUT

```
obscure@Obscures-MacBook-Air output % ./task_1
Enter a Number to check divisibility of: 9

You entered an ODD number (9)
9 is divisible by 3

obscure@Obscures-MacBook-Air output % ./task_1
Enter a Number to check divisibility of: 12

You entered an EVEN number (12)
12 is divisible by 4
12 is divisible by 6

obscure@Obscures-MacBook-Air output % ./task_1
Enter a Number to check divisibility of: 100000

You entered an EVEN number (100000)
100000 is divisible by 4
```

# Task 2 [CLO 1]:

## CODE

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

int main()
{
    float marks;
    char grade = ' ';

    std::cout << "Enter marks: ";
    std::cin >> marks;
    std::cout << "\n";

    if (marks >= 70)
        grade = 'A';
    else if (marks >= 60 && marks < 70)
        // even though condition `marks < 70` is not required
        // it is included to imporve readability
        grade = '+'; // assigning unique character
    else if (marks >= 50 && marks < 60)
        grade = 'B';
    else if (marks >= 40 && marks < 50)
        grade = 'C';
    else if (marks < 40)
        grade = 'F';

    // when printing convert '+' to "B+" (char -> std::string)
    // converting because string allows storing multiple characters
    std::string grade_printable;
    if (grade == '+')
        grade_printable = "B+";
    else
        grade_printable = grade;
    std::cout << marks << " marks equate to grade " << grade_printable << std::endl;

    switch (grade)
    {
    case 'A':
        std::cout << "Outstanding!" << std::endl;
        break;
    case '+':
        std::cout << "Excellent!" << std::endl;
        break;
    case 'B':
        std::cout << "Very Good!" << std::endl;
        break;
    case 'C':
        std::cout << "Good!" << std::endl;
        break;
    case 'F':
        std::cout << "Fail!" << std::endl;
        break;
    default:
        std::cout << "THIS WILL NEVER PRINT" << std::endl;
        break;
    }

    // ignore is used to clear previous inputs from user
    std::cin.ignore();
    std::cin.get();
    return 0;
}
```

## OUTPUT

```
obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 56

56 marks equate to grade B
Very Good!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 75

75 marks equate to grade A
Outstanding!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 65

65 marks equate to grade B+
Excellent!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 55

55 marks equate to grade B
Very Good!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 45

45 marks equate to grade C
Good!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 35

35 marks equate to grade F
Fail!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 25

25 marks equate to grade F
Fail!

obscure@Obscures-MacBook-Air output % ./task_2
Enter marks: 15

15 marks equate to grade F
Fail!
```

# Task 3 [CLO 1]:

## CODE

```cpp
#include <iostream>
#include <iomanip> // manipulators for printRow

float getPrice(int product_id);
void printRow(const char *colA, const char *colB);
void printProductTable();

int main()
{
    int product_id, quantity;
    float total = 0.f;
    float discount = 0.f;

    printProductTable();
    // loop to run until valid value for product id is entered by user
    while (true)
    {
        std::cout << "Enter Product ID: ";
        std::cin >> product_id;
        std::cout << "\n";

        if (getPrice(product_id) == -1)
        {
            std::cout << "Please input a valid value from the table";
            printProductTable();
            continue;
        }

        // break out of loop when value is validated
        break;
    }

    std::cout << "Enter Quantity: ";
    std::cin >> quantity;
    std::cout << "\n";

    total = getPrice(product_id) * quantity;

    // apply 10% discount is user buys more than 5 items
    if (quantity > 5)
        discount = total * (10.f / 100);

    std::cout << "Final amount owed: $" << total - discount << std::endl;

    // print discount info
    if (discount > 0)
    {
        std::cout << std::endl;
        std::cout << "Discount Applied: $" << discount << std::endl;
        std::cout << "Without Discount: $" << total << std::endl;
    }

    // ignore is used to clear previous inputs from user
    std::cin.ignore();
    std::cin.get();
    return 0;
}
```

```cpp
float getPrice(int product_id)
{
    switch (product_id)
    {
    // no break is needed as return does the job
    case 1:
        return 2.98f;
    case 2:
        return 4.5f;
    case 3:
        return 9.98f;
    case 4:
        return 4.49f;
    case 5:
        return 6.87f;

    default:
        return -1.f;
    }
}

void printRow(const char *colA, const char *colB)
{
    std::cout << " " << std::left << std::setw(4) << colA << " | " << colB << "\n";
}

// print a formatted table that shows all product;
void printProductTable()
{
    const char *LINES = "--------------";
    std::cout << "\n\n"
              << LINES << "\n";
    printRow("ID", "Price");
    std::cout << LINES << "\n";
    printRow("1", "$2.98");
    printRow("2", "$4.50");
    printRow("3", "$9.98");
    printRow("4", "$4.49");
    printRow("5", "$6.87");
    std::cout << LINES << "\n\n";
}
```

## OUTPUT

```
obscure@Obscures-MacBook-Air output % ./task_3


 ------------
  ID   | Price
 ------------
  1    | $2.98
  2    | $4.50
  3    | $9.98
  4    | $4.49
  5    | $6.87
 ------------

 Enter Product ID: 2

 Enter Quantity: 2

 Final amount owed: $9

obscure@Obscures-MacBook-Air output % ./task_3


 ------------
  ID   | Price
 ------------
  1    | $2.98
  2    | $4.50
  3    | $9.98
  4    | $4.49
  5    | $6.87
 ------------

 Enter Product ID: 4

 Enter Quantity: 5

 Final amount owed: $22.45

obscure@Obscures-MacBook-Air output % ./task_3


 ------------
  ID   | Price
 ------------
  1    | $2.98
  2    | $4.50
  3    | $9.98
  4    | $4.49
  5    | $6.87
 ------------

 Enter Product ID: 3

 Enter Quantity: 10

 Final amount owed: $89.82

 Discount Applied: $9.98
 Without Discount: $99.8
```

# Task 4 [CLO 1]:

## CODE

```cpp
#include <iostream>

int main()
{
    float annualSales;
    float bonus;

    std::cout << "Enter Annual Sales: $";
    std::cin >> annualSales;
    std::cout << "\n";

    if (annualSales >= 15'000)
    {
        bonus = annualSales * (2.f / 100);
    }
    else
    {
        bonus = annualSales * (1.5f / 100);
    }

    std::cout << "Bonus: $" << bonus << "\n";

    // ignore is used to clear previous inputs from user
    std::cin.ignore();
    std::cin.get();
    return 0;
}
```

## OUTPUT

```
obscure@Obscures-MacBook-Air output % ./task_4
Enter Annual Sales: $14999

Bonus: $224.985

obscure@Obscures-MacBook-Air output % ./task_4
Enter Annual Sales: $15000

Bonus: $300

obscure@Obscures-MacBook-Air output % ./task_4
Enter Annual Sales: $15001

Bonus: $300.02

obscure@Obscures-MacBook-Air output % ./task_4
Enter Annual Sales: $10000

Bonus: $150

obscure@Obscures-MacBook-Air output % ./task_4
Enter Annual Sales: $20000

Bonus: $400

obscure@Obscures-MacBook-Air output %
```

# Task 5 [CLO 1]:

## CODE

```cpp
#include <iostream>

void printCase(char a)
{
    // WORKING PRINCIPLE
    // char is basically be thought of as a unsigned 8 bit integer
    // the computer stores a map from each possible number 0-255 to a specific
symbol
    // in this map, small, capital, digits etc are sequential, comes after one
another
    // by using only (char) >= (char) both are implicitely converted to int and
compared
    // as said since digits are mapped sequentially the number representation of a
small alphabet is between the number representation of `a` and `z`
    // same holds for capital letters and digits
    if (a >= (int)'a' && a <= (int)'z')
        std::cout << "You entered SMALL letter \'" << a << "\'";
    else if (a >= 'A' && a <= 'Z')
        std::cout << "You entered CAPITAL letter \'" << a << "\'";
    else if (a >= '0' && a <= '9')
        std::cout << "You entered DIGIT \'" << a << "\'";
    else
        std::cout << "You entered SPECIAL character \'" << a << "\'";
    // (a >= 'a' && a <= 'z') can also be written as
    // (a >= (int)'a' && a <= (int)'z')
    // above is not used as this clutters the code
    std::cout << "\n";
}

int main()
{
    char inputChar;

    std::cout << "Enter a character: ";
    std::cin >> inputChar;
    std::cout << "\n";

    printCase(inputChar);

    // ignore is used to clear previous inputs from user
    std::cin.ignore();
    std::cin.get();
    return 0;
}
```

## OUTPUT

```
obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: a

You entered SMALL letter 'a'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: z

You entered SMALL letter 'z'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: g

You entered SMALL letter 'g'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: A

You entered CAPITAL letter 'A'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: Z

You entered CAPITAL letter 'Z'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: G

You entered CAPITAL letter 'G'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: 0

You entered DIGIT '0'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: 9

You entered DIGIT '9'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: 5

You entered DIGIT '5'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: !

You entered SPECIAL character '!'

obscure@Obscures-MacBook-Air output % ./task_5
Enter a character: +

You entered SPECIAL character '+'
```