

Lab#1: Getting Started with Computer Solving Problems

Objectives:

The main objective of this lab is to review some fundamental concepts related to digital computers and to practice students with the analysis and the design (making algorithms and flowcharts) for some computer solving problems.

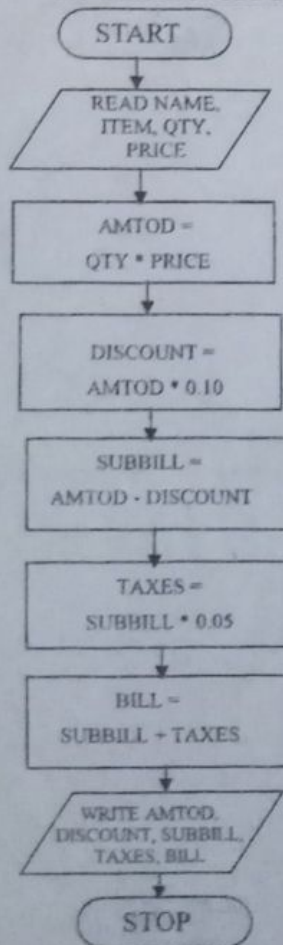
PA1/T1: Introduction to Computers and Computing

Review Questions:

1. What is computer science?
2. What is a digital computer?
3. Write full forms of the following: PC, CPU, ALU, RAM, ROM.
4. Briefly describe the functions of the different hardware components of a conventional digital computer with the help of a suitable block diagram.
5. What is a CPU? What is its function?
6. Explain both the primary and the secondary memory units.
7. What is a computer program?
8. What are the three main software categories?
9. Briefly state the role of the operating system in a computer system.
10. How software applications are produced?

PA1/T2: Computer Solving Problems (Analysis and Design)

1. Write an algorithm and draw a flowchart that will read the two sides of a rectangle and calculate its area.
2. Write an algorithm and draw the flowchart for finding the area and circumference of a circle.
3. Look at the flowchart below and answer the following questions:
 - (a) Which problem this flowchart is used to solve?
 - (b) What are the inputs/outputs used to solve that problem?
 - (c) Simulate the execution of this flowchart and complete the table below, assuming the values shown in the table are read as inputs.



NAME	Mr. R. Namane
ITEM	Engineering Book
QTY	3
PRICE	49.99
AMTOD	149.97
DISCOUNT	14.997
SUBBILL	134.973
TAXES	6.74865
BILL	141.72165

**Lab#2: More on Writing Algorithms and Drawing Flowcharts
(Sequence, Decision, and Repetition features)**

Objectives:

The main objective of this lab is to practice students more with the analysis and the design (making algorithms and flowcharts) for some computer solving problems. We focus more on the use of decision and repetition constructs in making algorithms and flowcharts.

Problem 1:

Write an algorithm and draw a flowchart that prints out the biggest of two inputted numbers.

Problem 2:

Repeat problem 1 for three inputted numbers.

Problem 3:

Write an algorithm and draw a flowchart that calculates the roots of a quadratic equation $ax^2 + bx + c = 0$.

Remember that: $d = (b^2 - 4ac)$, and the roots are: $x_1 = (-b + \sqrt{d})/2a$ and $x_2 = (-b - \sqrt{d})/2a$.

Problem 4:

Write an algorithm and draw a flowchart that reads a number and then checks if it is between 10 and 15. If it is, that number will be displayed otherwise a message "out of range" will be displayed instead.

Problem 5:

Write an algorithm and draw a flowchart that prints out "Hello World" on the screen hundred times.

Problem 6:

Write an algorithm and draw a flowchart that calculates the addition of ten inputted random numbers. The numbers are entered one by one. When the last one is entered, their summation will be displayed.

Problem 7:

Repeat problem 6 but now instead of having ten numbers, the size of the list is specified first by the user. When the last number is entered, the summation of all previously entered numbers will be displayed.

Problem 8:

Again repeat problem 6 but now instead of having ten numbers, the size of the list is unknown and the process of reading numbers is stopped when the user enters "-1". When "-1" is entered, the summation of all previously entered numbers will be displayed.

Problem 9:

Write an algorithm and draw a flowchart to ask for a number. That number should be less than 50 and If not the user is asked again to re-enter another one. Once a correct number is read, keep adding 1 to it until 50 is reached.

Problem 10:

Develop the algorithm and draw a flowchart for finding the sum of the series $1 + 2 + 3 + 4 + \dots$ up to N terms.

Problem 11:

Write an algorithm and draw a flowchart for determining the sum of the series $2 + 4 + 8 + 16 + \dots$ up to N.

Problem 12:

Write an algorithm and draw a flowchart for calculating the factorial of a given number N.

Problem 13:

Write an algorithm to find the sum of the series $1 + x + x^2 + x^3 + x^4 + \dots$ up to x^n terms.

Problem 14:

Write an algorithm for computing the sum of the series $1 + x^2/2! + x^3/3! + x^4/4! \dots$ up to N terms.

Handwritten note: $(3+1) = 4$
 $1+2+3+6$
 $6/2 = 3$

Lab#3: Computer Solving Problems (Analysis and Design)**Objectives:**

The main objective of this lab is to make students practice more the analysis and the design (making algorithms and flowcharts) of computer solving problems. Students will be requested to analyze and then show how to best solve some general problems using digital computers by creating algorithms and flowcharts.

Problem#1: Calculating Daily Salary

A company pays its employees on a task-based principle. Create an algorithm and flowchart that determines the daily salary of an employee such that:

- The employee is paid for each piece realized. The amount given for each piece is AP.
- The total number of correct pieces realized per day is NCP:
 - If $NCP \leq 100$, the employee receives $NCP * AP$
 - If $NCP > 100$, the employee receives $120 * AP$
- The company removes 10% of the daily salary for social security charges.

Your solution should print out the total daily salary, the social security charges, and the net daily salary (without charges).

Problem#2: Distance Dropped Problem

Create an algorithm and flowchart that calculates and prints the distance traveled each second by an object dropped from a height in a standard gravitational field beginning at $t = 0$ sec over the next 10 seconds.

Problem#4: Population Growth Rate

The annual growth rate of an "X" country is 1% for a population that has reached 50 million for the current year. It is 5% for a country "Y" with a population of 40 million for the same year.

Create an algorithm and flowchart that finds in how many years the population of country "Y" will exceed that of country "X".

Problem#3: Calculating the Value of π

Create an algorithm and flowchart that calculates the value of π from the infinite series:

$$\pi = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \frac{4}{11} + \dots$$

The user will be asked first how many terms of this series does he have to use before printing the approximated value of π .

Lab#4: Getting Started with Code Blocks IDE

Objectives:

- Introduce students to C programming.
- Write, compile and execute C programs using Code Blocks IDE.

Introduction:

To get started with C programming, I suggest using a simple IDE development environment. **CodeBlocks** is a nice IDE with a free C/C++ compiler and which supports both Windows and Linux platforms. CodeBlocks can be downloaded and installed on your own laptop/PC from the link: <http://www.codeblocks.org/>

Important Remarks:

- One or two students per computer station are allowed (no more than two students for whatever reason).
- For each laboratory session, the first thing to do is to create a new folder named **Gxlab4** (where "x" is your group number and "y" is the number of that lab session, for example if you are from group one then call it **G1lab4**) which will be used to save the work of all parts of your laboratory experiment.
- At the end of your laboratory session, delete your folders **Gxlab4**, shut down your computers, and clean your stations.
- Students working on the same computer station must submit a laboratory report for their experiment within one week. Your lab instructor will provide you with the content and the way you present your report.
- Note that your pre-lab work (what is done before lab), in-lab work (what is done during the lab), and post-lab (lab report) are all considered in the laboratory grading policy.

Part 1: My first program in C

In CodeBlocks, a program is created by creating a new project. Therefore, start CodeBlocks and create a new project. To create a project, click on the **File** pull-down menu, open **New** and then **Project**. This will bring up the **New from template** window shown in fig. 1. In the dialog window, select "**Console Application**" and click **Go**.

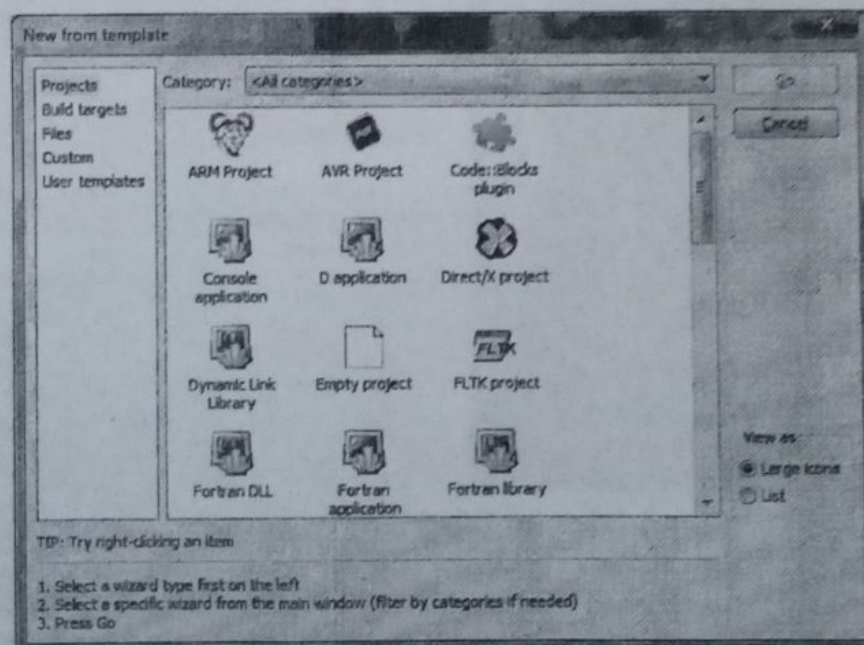



Fig.1

Then select **C** as the type of language you want to use and click **Next** (fig. 2). Then, type **part1** as the Project title, and use your **Gxlab4** folder (you have already created) as the folder to create the project in. To specify the location of the folder to contain the project, click on the  button in fig. 3 and browse to your **Gxlab4** folder to store the project. Click **Next** (fig. 3) and then click **Finish** (fig. 4).

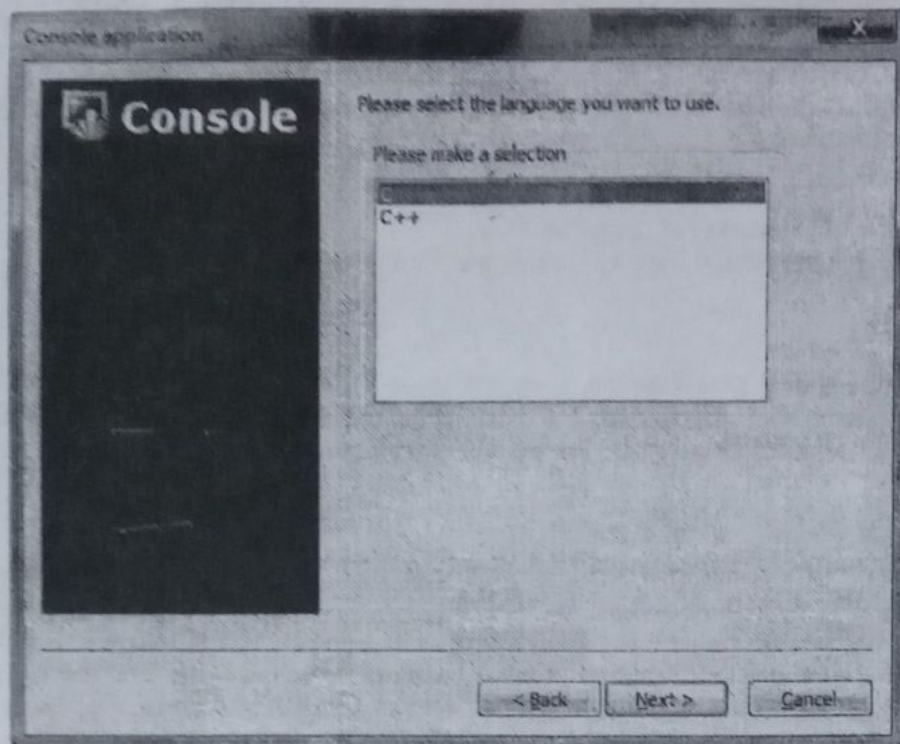


Fig. 2

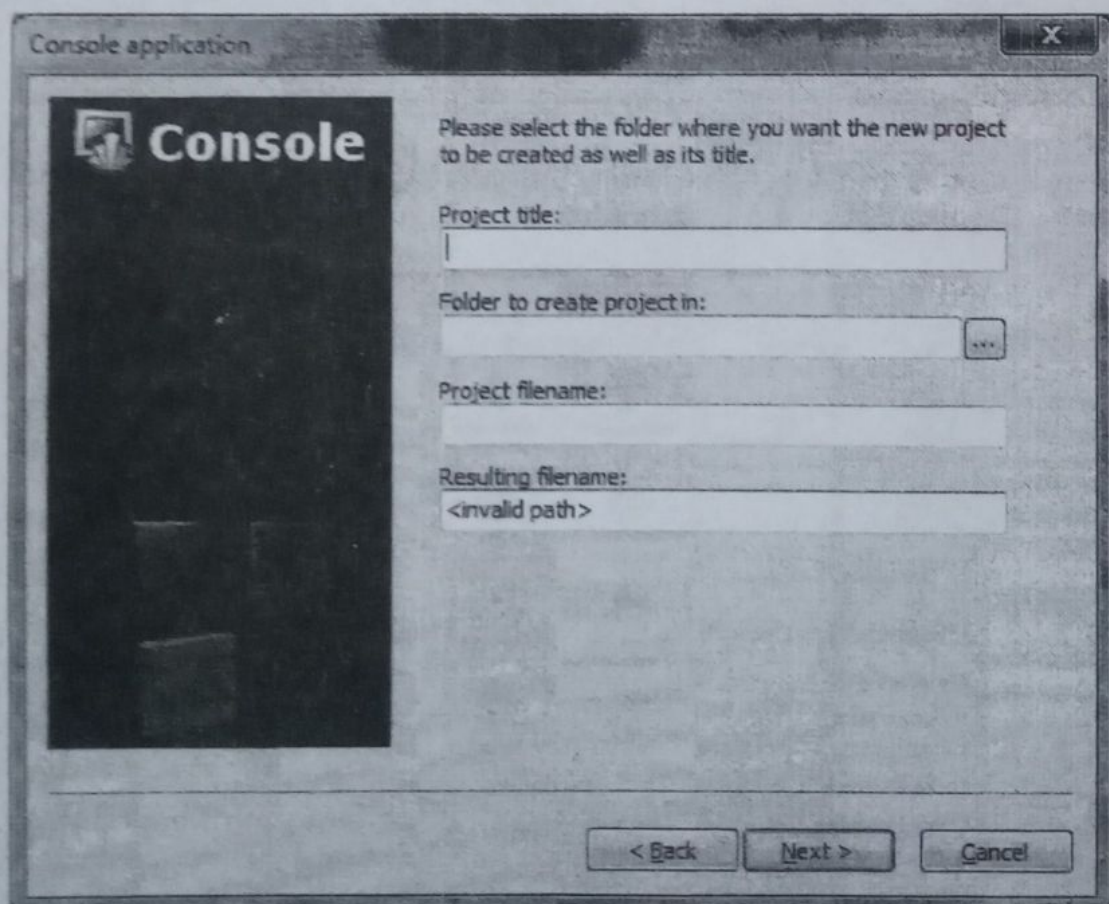


Fig. 3

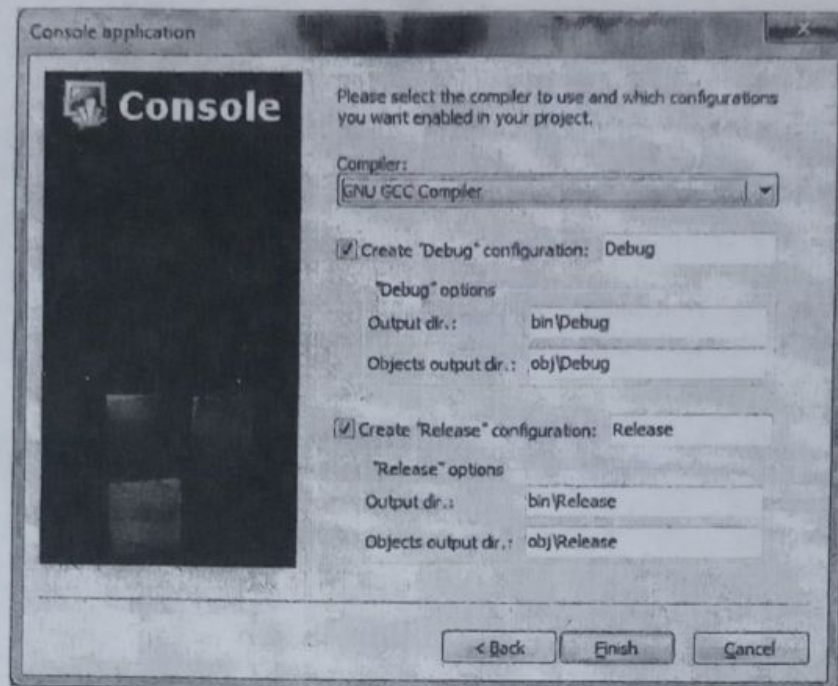



Fig. 4

This will automatically create a simple **main.c** template file (fig. 5). Replace the content of the **main.c** file with the code mentioned in fig. 6 and then save your work by clicking on the  icon of the bar menu.

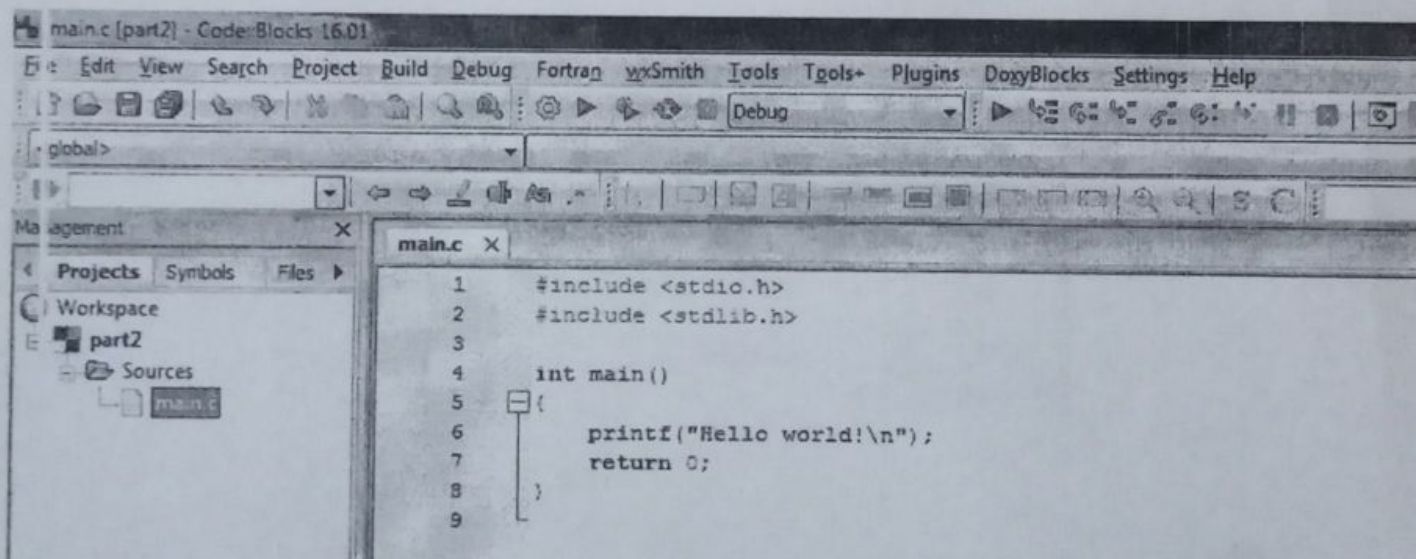


Fig. 5

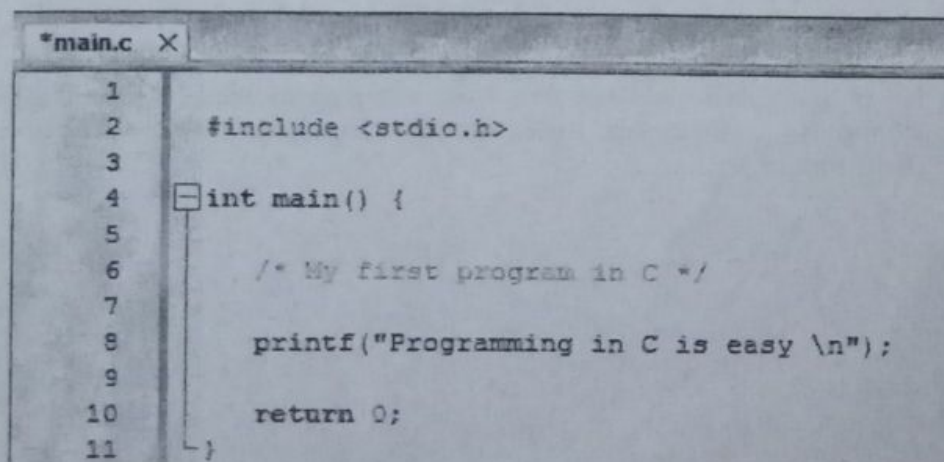


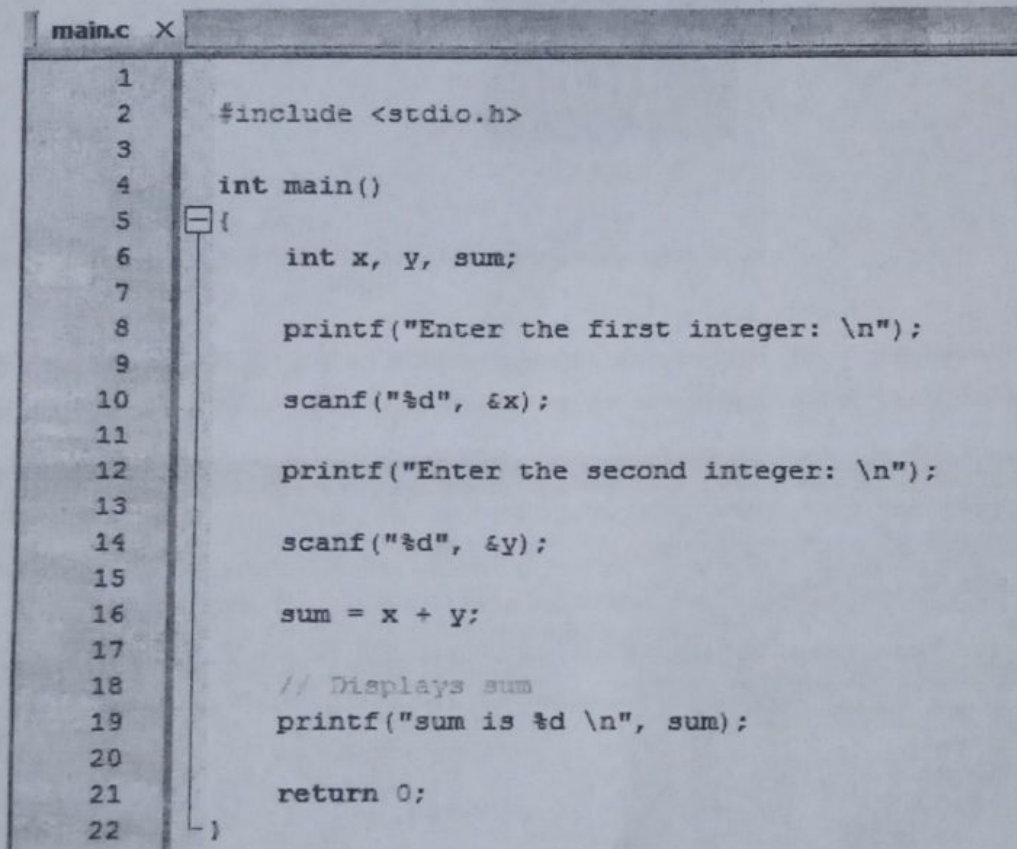


Fig. 6

To compile this program, you can use the **Build** pull-down menu and click on **Build**. There is also an icon on the bar menu  that help you to compile directly your program. If any error is highlighted, correct it first before proceeding further. To execute your program, go to **Build->Run** or click on the run icon  in the bar menu. This should execute your program, which simply prints at the prompt "Programming in C is easy". Press any key of your keyboard to leave the prompt.

Part 2:

Create another project called part2 and then enter the code shown in fig. 7. Compile and then run your program. Test your program for several inputs (both positive and negative numbers should be used) to make sure that it functions correctly.



```
1
2  #include <stdio.h>
3
4  int main()
5  {
6      int x, y, sum;
7
8      printf("Enter the first integer: \n");
9
10     scanf("%d", &x);
11
12     printf("Enter the second integer: \n");
13
14     scanf("%d", &y);
15
16     sum = x + y;
17
18     // Displays sum
19     printf("sum is %d \n", sum);
20
21     return 0;
22 }
```

Fig. 7

Part 3:

Create another project called part3 and then write a C program that will read the two sides of a rectangle and calculate its area. Compile and then execute your program. Test your program for several values of both the length and the width of your rectangle to make sure that it functions correctly.

Part 4:

Create a last project called part4 and then write a C program that finds the area and circumference of a circle. Compile and then run your program. Again, test your program for several values of the circle's radius to make sure that it functions correctly.

Lab#5: Constant, Variables, and Data types**Objectives:**

The main objective of this lab is to practice more the use of variables, constants, and data types in C. Students will be asked to run several C programs related to this topic and to analyze their outputs.

1. Consider the following program. Compile and run it. What is the output? Is this what you expected?

Why?

```
#include <stdio.h>
int main() {
    int number;
    printf("%d\n", number);
    return 0;
}
```

2. Consider the following program. Compile and run it. What is the output?

```
#include <stdio.h>
int main()
{
    const float PI = 3.14;
    float rad = 3.5;

    printf("Circle area = %.2f\n", PI*rad*rad);
    printf("Circle circumference = %.2f\n", 2*PI*rad);
    printf("Circle diameter = %.2f\n", 2*rad);
    return 0;
}
```

Now modify the previous program as follows and then recompile it gain (use copy/past for the three printf() statements). What happens? Remove the keyword const from the PI declaration and then recompile the program. What you notice?

```
#include <stdio.h>
int main()
{
    const float PI = 3.14;
    float rad = 3.5;

    printf("Circle area = %.2f\n", PI*rad*rad);
    printf("Circle circumference = %.2f\n", 2*PI*rad);
    printf("Circle diameter = %.2f\n", 2*rad);
    PI = 3.14 + 5.0;
    printf("\nCircle area = %.2f\n", PI*rad*rad);
    printf("Circle circumference = %.2f\n", 2*PI*rad);
    printf("Circle diameter = %.2f\n", 2*rad);

    return 0;
}
```

3. Consider the following program. Compile and run it. What is the output? What can you remark?

```
#include <stdio.h>
int main()
{
    float p = 0.0;
    int r = 1, s = 2;

    p = r/s;
    printf("\np = r/s = %.2f\n", p);
    return 0;
}
```

Now modify the previous program as follows and then recompile it gain. What happens? What you conclude about the use of (float) when calculating p?

```
#include <stdio.h>
int main()
{
    float p = 0.0;
    int r = 1, s = 2;

    p = (float)r/s;
    printf("\np = (float)r/s = %.2f\n", p);
    return 0;
}
```

4. Write a C program to print following pattern. Use one printf() statement for each line of output. End each line by printing a newline (\n).

```
* * * * *
 * * * * *
* * * * *
 * * * * *
* * * * *
```

Repeat the same thing for the following two patterns.

```
* * * * *      * * * * *
*           *      *      *
*           *      *      *
*           *      *      *
* * * * *      *
```

5. Write a program to prompt user for 3 real numbers and print their product. Use a float variable *product* to store the product. Display your result with 3 digits of precision.
6. Write a program in which you declare eight characters and initialize them with 'p', 'r', 'o', 'g', 'a', 'm', 'i', and 'n' respectively. Then use a printf() to display the word "programming" on the screen by concatenating those declared characters.

Lab#6: Operators, Conditional and Iterative Constructs.**Objectives:**

The main objective of this lab is to enhance student's familiarity with C programming by applying conditional and iterative constructs for solving some given problems.

Problem 1:

Find the value that is assigned to the variables x, y, and z when the following program is executed.

```
int main()
{
    int x, y, z;
    x = 2 + 3 - 4 + 5 - (6 - 7);
    y = 2 * 33 + 4 * (5 - 6);
    z = 2 * 3 * 4 / 15 % 13;
    x = 2 * 3 * 4 / (15 % 13);
    y = 2 * 3 * (4 / 15 % 13);
    z = 2 + 33 % 5 / 4;
    x = 50 % (5 * (16 % 12 * (17/3)));
    Y = -2 * 3 % 4 / 5 - 6 + 7;
    z = 8 / 4 / 2 * 2 * 4 * 8 % 13 % 7 % 3;
    return 0;
}
```

By inserting appropriate calls to printf(), verify the answers obtained.

Problem 2:

Write a C program to find largest number (between 3) using if-else statement.

Problem 3:

Write a C program for solving a quadratic equation $ax^2+bx+c=0$ and find the roots of x.

Test your program for the following three examples:

- $x^2+2x-8=0$: $x_1=2$, $x_2=-4$.
- $x^2-10x+25=0$: $x=5$.
- $5x^2-2x+2=0$: no real roots.

Problem 4:

Write a C program to generate the electricity bill according to their consumption based on both the old and the new index of electricity.

Total Consumption in KWh per quarter	Rate per unit
0 ~ 125	1.77 DA
125 ~ 250	4.17 DA
250 ~ 1000	4.81 DA
1000 ~ above	5.48 DA

Problem 5:

Using for loop, write a C program to print the sum of the series $1+2+3+4+\dots$ up to n terms.

Problem 6:

Write a C program to print the sum of the following series up to n terms where n is given by the user.

$1 + x + x^2 + x^3 + \dots$ (The value of x is also given by the user).

Problem 7:

Write a C program to print the following triangle.

```
1
2 2
3 3 3
4 4 4 4
5 5 5 5 ... up to nth line
```

Problem 8:

Write a C program that prompts the user to enter the date as three integer values for the month, the day in the month, and the year. The program should then output the date in the form 31 December 2017 when the user enters, say 12 31 2017. Use switch statement to display the appropriate month name.

Problem 9:

Computer shop makes laptops for sale with 400\$ per item. The shop provides a discount rate for customers on their total purchase price (TP) as follows:

- When $TP \leq 1000\$$, the discount rate is 0%.
- When $1000\$ < TP < 2000\$$, the discount rate is 5%.
- When $TP \geq 2000\$$, the discount rate is 10%.

Write a C program to read the number of items selected by the customer and then printing out the total purchase price TP (without discount) and the discounted price DTP.