



**Module Title**

**Principles of Programming**

**Weekly Assignment - Practical 3**

**Year**

**2025**

**Student Name: NIRVIK K.C.**

**UWE ID: 25024649**

**Assignment Submission Date: January 3, 2026**

This assignment consists of the programming questions from practical exercises related to the topics from week 3.

## Questions

1. Write, compile and run a C program to read from an input file.

Given code :

```
/* read values from input file Practical 3, Part 2
@author your name */
#include FILE *fp;
// define a file pointer for the file to hold a disk location
int main()
{
    // open the file and assign its address/disk location to file pointer
    fp = fopen("inputFile.txt", "r"); //relative pathname used
    // add rest of program here – use fp for the file
    fclose (fp); // always close files you've opened
}
```

**Answer:**

Step 1: Create a folder(practical3) to store the program in this practical.

Step 2: Create a text file(inputFile.txt).

Step 3: Use the text file(inputFile.txt) as input file.

Step 4: Reading from the input file.

Step 5: Run the program

**Following code for input:**

/\* read values from input file

Practical 3, Part 1 (a)

@Nirvik K.C. \*/

```
#include <stdio.h>
```

```
FILE *fp; // define a file pointer for the file to hold a disk location
```

```
int main()
```

```
{
```

```
// open the file and assign its address/disk location to file pointer  
fp = fopen("inputFile.txt", "r"); // relative pathname used  
  
// declare variables for holding the values of input  
  
char firstWord[20];  
  
char secondWord[20];  
  
int num;  
  
printf("Reads two words and an integer from file \n");  
  
// Read two words and an integer from file  
  
// fscanf instead of scanf used, and file pointer needed  
fscanf(fp, "%s %s %d", firstWord, secondWord, &num);  
  
// display two words and an integer  
  
printf("Displays back what has been read from input file:\n");  
  
printf("%s %s \n%d \n", firstWord, secondWord, num);  
  
fclose(fp); // Closes the file  
  
return 0;  
}
```

**/\* Output :**

Reads two words and an integer from file

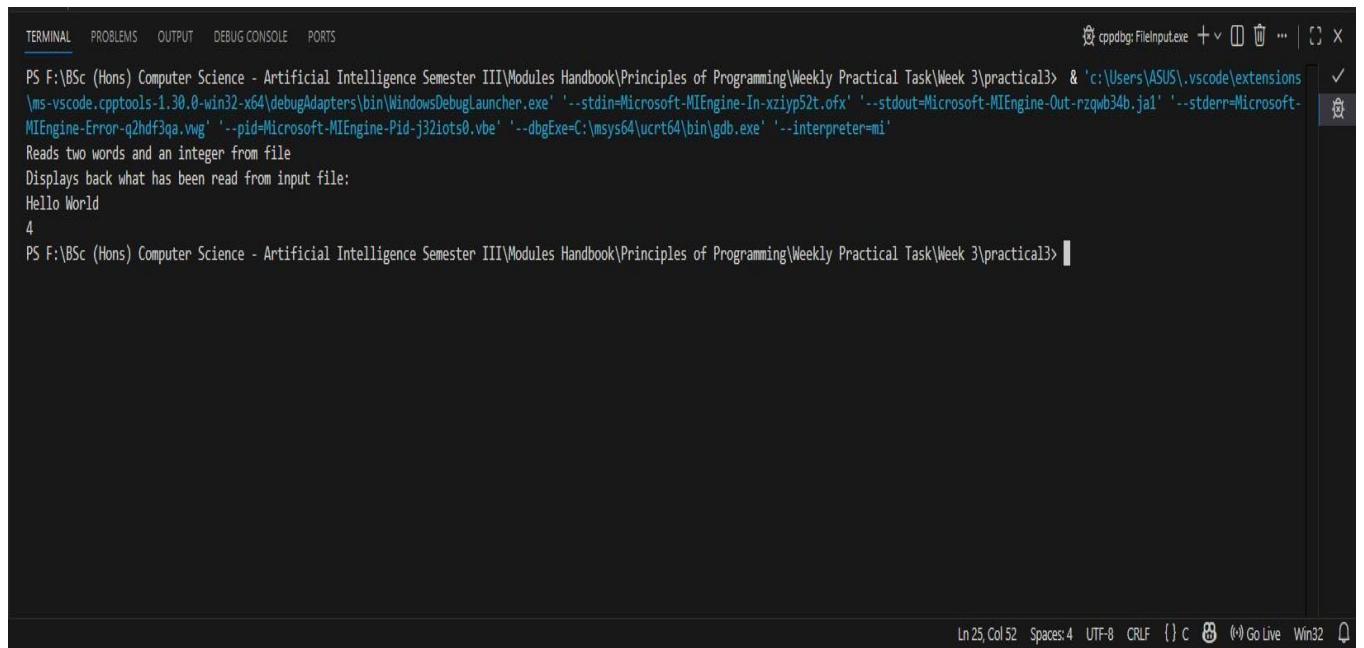
Displays back what has been read from input file:

Hello World

4

\*/

### Output obtained in execution:



```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg:FileInput.exe + v □ ⌂ ⌂ X
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-xziyp52t.ofx' '--stdout=Microsoft-MIEngine-Out-rzqwb34b.ja1' '--stderr=Microsoft-MIEngine-Error-q2hdf3qa.wvg' '--pid=Microsoft-MIEngine-Pid-j32iots0.vbe' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Reads two words and an integer from file
Displays back what has been read from input file:
Hello World
4
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

Ln 25, Col 52 Spaces:4 UTF-8 CRLF {} C ⌂ ⓘ Go Live Win32

1. (b) Change the layout of the values in the input file with the three input values on the same line and check whether your program still works.

**Answer:**

**Following code for input:**

```
/* read values from input file
```

Practical 3, Part 1 (b)

@Nirvik K.C. \*/

```
#include <stdio.h>

FILE *fp; // define a file pointer for the file to hold a disk location

int main()
{
    FILE *fp;
    // open file for reading
    fp = fopen("inputFile.txt", "r");
    if (fp == NULL)
    {
        printf("Error opening the file");
        return 1;
    }
}
```

```
// declare variables for holding the values of input
char firstWord[20];
char secondWord[20];
int num;
printf("Reads two words and an integer from file \n");
// Changed the layout of the values in the input file - Hello World 4
// Read two words and an integer from file
fscanf(fp, "%s %s %d", firstWord, secondWord, &num);
// Display two words and an integer
printf("Displays back what has been read from input file:\n");
printf("%s %s %d \n \n", firstWord, secondWord, num); // Display the three
input values on the same line
fclose(fp); // Closes the file
return 0;
}

/* Output :
Reads two words and an integer from file
Displays back what has been read from input file:
Hello World 4
*/
```

## Output obtained in execution:

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg:FileInput2.exe + v ⚡ ... | X
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:/Users/ASUS/.vscode/extensions/ms-vscode.cpptools-1.30.0-win32-x64/debugAdapters/bin/WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-52a501xm.gcc' '--stdout=Microsoft-MIEngine-Out-t4p03spv.rsw' '--stderr=Microsoft-MIEngine-Error-cwjyvo2d.f4u' '--pid=Microsoft-MIEngine-Pid-4qvtm1uj.ts5' '--dbgExe=C:/msys64ucrt64/bin/gdb.exe' '--interpreter=mi'
Reads two words and an integer from file
Displays back what has been read from input file:
Hello World 4

PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

Ln 26 Col 74 Spaces:4 UTF-8 CRLF () C (Go Live Win32)

2. Create a text file and save it in the practical3 folder. Name the file as bill\_input.txt, which contains one set of input values only as follows:  
900 800 23 5  
Save ElectricityBillA.c in the practical2 folder as ElectricityBillB.c in the same folder. Program should read the input data from input file billInput.txt instead of reading the input data from the keyboard. Add a loop into your program so that your program will run only once by looping through the 7 sets of values.

Given code:

```
//loop begins  
for (int i = 1; i <= 7; i++)  
{  
    //Adapt the existing block of code for reading and validating one set of input  
    //values (i.e. use fscanf with a file pointer)  
}  
//loop ends
```

Update the input file, bill\_input.txt, with the following input data, save it, run it, and test it :

```
900 800 23 5  
9000 9999 12 2  
9999 10005 14 6  
500 6000 26 7  
10000 10100 10 9  
6000 6890 30 2  
590 829 31 9
```

### Answer:

Following code for input:

```
/* read values from input file
```

Practical 3, Part 2

@Nirvik K.C. \*/

```
#include <stdio.h>

FILE *fp; // define a file pointer for the file to hold a disk location

int main()

{

    // open file for reading

    fp = fopen("bill_input.txt", "r");

    if (fp == NULL)

    {

        printf("Error opening the file");

        return 1;

    }

    // Declare variables to store input values

    int previous, current, day, month;

    int units;

    printf("Starting validation of electricity bill readings from the file\n\n");

    // Loop to read set of input values from the file

    for (int i = 1; i <= 7; i++)
```

```
{  
    printf("The Electricity Bill Set %d\n", i);  
  
    fscanf(fp, "%d %d %d %d", &previous, &current, &day, &month);  
  
    // Display the read values  
  
    printf("Input read: Previous Reading= %d, Current Reading= %d, Day= %d,  
Month= %d\n", previous, current, day, month);  
  
    // Validation checks  
  
    if (current < 0 || current > 9999)  
    {  
        printf("Error: Current meter reading out of range (0-9999)\n");  
    }  
  
    if (previous < 0 || previous > 9999)  
    {  
        printf("Error: Previous meter reading out of range (0-9999)\n");  
    }  
}
```

25024649

```
if (current - previous > 1000)
{
    printf("Error: Electricity used more than 1000 units\n");
}

if (previous > current)
{
    printf("Error: Previous reading is greater than current reading\n");
}

if (month < 1 || month > 12)
{
    printf("Error: Month out of range (1-12)\n");
}

// Check days in month

if ((month == 1 || month == 3 || month == 5 || month == 7 || month == 8 ||
month == 10 || month == 12) && (day < 1 || day > 31))
```

25024649

```
{  
    printf("Error: Day out of range for the given month (1-31)\n");  
}  
  
if ((month == 4 || month == 6 || month == 9 || month == 11) && (day < 1 ||  
day > 30))  
  
{  
    printf("Error: Day out of range 1 to 30\n");  
}  
  
if (month == 2 && (day < 1 || day > 29))  
  
{  
    printf("Error: Day out of range 1 to 29\n");  
}  
  
printf("\n");  
  
}  
  
fclose(fp); // Closes the file  
  
return 0;  
}
```

25024649

/\* Output :

Starting validation of electricity bill readings from the file

The Electricity Bill Set 1

Input read: Previous Reading= 900, Current Reading= 800, Day= 23, Month= 5

Error: Previous reading is greater than current reading

The Electricity Bill Set 2

Input read: Previous Reading= 9000, Current Reading= 9999, Day= 12, Month= 2

The Electricity Bill Set 3

Input read: Previous Reading= 9999, Current Reading= 10005, Day= 14, Month= 6

Error: Current meter reading out of range (0-9999)

The Electricity Bill Set 4

Input read: Previous Reading= 500, Current Reading= 6000, Day= 26, Month= 7

Error: Electricity used more than 1000 units

The Electricity Bill Set 5

Input read: Previous Reading= 10000, Current Reading= 10100, Day= 10, Month= 9

Error: Current meter reading out of range (0-9999)

Error: Previous meter reading out of range (0-9999)

## The Electricity Bill Set 6

Input read: Previous Reading= 6000, Current Reading= 6890, Day= 30, Month= 2

Error: Day out of range 1 to 29

## The Electricity Bill Set 7

Input read: Previous Reading= 590, Current Reading= 829, Day= 31, Month= 9

Error: Day out of range 1 to 30

\*/

## Output obtained in execution:

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '-stdin=Microsoft-MIEngine-In-g5xlxxv.vml' '--stdout=Microsoft-MIEngine-Out-on4neyp2.wqv' '--stderr=Microsoft-MIEngine-Error-agonyppw.wm0' '--pid=Microsoft-MIEngine-Pid-g0ay3ryb.afk' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mii'
Starting validation of electricity bill readings from the file

The Electricity Bill Set 1
Input read: Previous Reading= 900, Current Reading= 800, Day= 23, Month= 5
Error: Previous reading is greater than current reading

The Electricity Bill Set 2
Input read: Previous Reading= 9000, Current Reading= 9999, Day= 12, Month= 2
Error: Current meter reading out of range (0-9999)

The Electricity Bill Set 3
Input read: Previous Reading= 9999, Current Reading= 10005, Day= 14, Month= 6
Error: Current meter reading out of range (0-9999)

The Electricity Bill Set 4
Input read: Previous Reading= 500, Current Reading= 6000, Day= 26, Month= 7
Error: Electricity used more than 1000 units

The Electricity Bill Set 5
Input read: Previous Reading= 10000, Current Reading= 10100, Day= 10, Month= 9
Error: Current meter reading out of range (0-9999)
Error: Previous meter reading out of range (0-9999)

The Electricity Bill Set 6
Input read: Previous Reading= 6000, Current Reading= 6890, Day= 30, Month= 2
Error: Day out of range 1 to 29

The Electricity Bill Set 7
Input read: Previous Reading= 590, Current Reading= 829, Day= 31, Month= 9
Error: Day out of range 1 to 30

PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

3. An electricity company keeps the metre readings of their customers in a text file, which should consist of an integer (which is the number of lines in the file) followed by a series of lines each of which consists of a customer number, the previous metre reading and the current metre reading (separated by a space). A program is required to take the text file as an input file, read a set of input values from the input file at a time, calculate the number of units the customer has used over the period, classify the user as follows:

Above 1000 units – heavy user

500 – 1000 units – regular user

Below 500 units – light user

The program should produce statistics for each category of users and display the statistics on the screen.

For example,

Heavy users 20

Regular users 105

Light users 54

Step 1: Create a text file named testInput.txt, and save it to the practical3 folder.

The text file should contain following input data:

5

1078 9000 9999

1234 10000 10100

1134 6000 6890

4511 590 829

8122 500 2000

Step 2: Now, create a C program named CustomerStatistics.c in the same folder.

Step 3: Also, produce a program outline/pseudocode before creating the C program.

Step 4: Write the C program to implement your program outline.

Step 5: Run and test your program.

**Answer:**

**Following pseudocode:**

START

Open testInput.txt for reading

IF file cannot be opened

    Display "Error: Cannot open file testInput.txt"

    Stop program

END IF

Read number\_of\_customers from file

Set heavyCount = 0

Set regularCount = 0

Set lightCount = 0

FOR i = 1 to number\_of\_customers

    Read customer\_number, previous\_reading, current\_reading

    unitsUsed = current reading - previous reading

    IF unitsUsed > 1000

        heavyCount = heavyCount + 1

25024649

ELSE IF unitsUsed >= 500

    regularCount = regularCount + 1

ELSE

    lightCount = lightCount + 1

END FOR

**Following code for input:**

/\* read values from input file

Practical 3, Part 3

@Nirvik K.C. \*/

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
FILE *fp;
```

```
int total_customers, customer_num, previous_read, current_read;
```

```
int units_used;
```

```
int heavy_users = 0;  
  
int regular_users = 0;  
  
int light_users = 0;  
  
fp = fopen("testInput.txt", "r");  
  
if (fp == NULL)  
  
{  
    printf("Error opening the file");  
  
    return 1;  
  
}  
  
fscanf(fp, "%d", &total_customers);  
  
int i;  
  
for (i = 0; i < total_customers; i++)  
  
{  
    fscanf(fp, "%d %d %d", &customer_num, &previous_read, &current_read);  
  
    units_used = current_read - previous_read;
```

```
if (units_used > 1000)
{
    heavy_users++;
}

else if (units_used >= 500)
{
    regular_users++;
}

else if (units_used < 500)
{
    light_users++;
}

else
{
    printf("Meter reading error for customer %d\n", customer_num);
}
```

25024649

```
printf("Customer Electricity Usage Statistics:\n");

printf("Heavy users: %d\n", heavy_users);

printf("Regular users: %d\n", regular_users);

printf("Light users: %d\n", light_users);

fclose(fp);

return 0;

}

/* Output:

Customer Electricity Usage Statistics:

Heavy users: 1

Regular users: 2

Light users: 2

*/
```

## Output obtained in execution:

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg:CustomerStatistics.exe + ⌂ ⌂ ... | ⌂ X
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-opeon54b.4tb' '--stdout=Microsoft-MIEngine-Out-fwfqv50n.azq' '--stderr=Microsoft-MIEngine-Error-mpch1myb.llx' '--pid=Microsoft-MIEngine-Pid-kciyag01.p2' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Customer Electricity Usage Statistics:
Heavy users: 1
Regular users: 2
Light users: 2
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

Ln 51, Col 2 Spaces:4 UTF-8 CRLF {} C ⌂ ( ⓘ Go Live Win32 ⌂

## 4. More Programming exercises

Use nested loops that display the following patterns in four separate programs:

### 4. (a) Display Pattern A:

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

1 2 3 4 5 6

**Answer:**

**Following code for input:**

```
/* read values from input file
```

Practical 3, Part 4 - 3.1, Display Pattern A

@Nirvik K.C. \*/

```
#include <stdio.h>

int main()
{
    int i, j, rows;
    // Prints number of lines (rows of numbers)
    printf("\nEnter the number of rows: ");
    scanf("%d", &rows);
    for (int i = 1; i <= rows; i++) // for number of rows
    {
        for (int j = 1; j <= i; j++) // for printing numbers in each row
```

```

{
    printf("%d ", j);

}

printf("\n"); // go to next line after printing each row

}

return 0;
}

```

### Output obtained in execution:

```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg: Display/PatternA.exe + v [ ] x

PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-rxcjimq2.nwg' '--stdout=Microsoft-MIEngine-Out-vvxvum0f.va4' '--stderr=Microsoft-MIEngine-Error-l1qccjje.hzc' '--pid=Microsoft-MIEngine-Pid-32edqteh.o25' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'

Enter the number of rows: 6
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

4. (b) Display Pattern B

1 2 3 4 5 6

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

**Answer:**

**Following code for input:**

```
/* read values from input file
```

Practical 3, Part 4 - 3.1, Display Pattern B

@Nirvik K.C. \*/

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int i, j, rows;
```

```
    // Prints number of lines (rows of numbers)
```

```

printf("\nEnter the number of rows: ");

scanf("%d", &rows);

for (i = rows; i >= 1; i--) // for number of rows

{
    for (j = 1; j <= i; j++) // for printing numbers from 1 to current row value (i)

    {
        printf("%d ", j);

    }

    printf("\n"); // go to next line after printing each row
}

return 0;
}

```

## Output obtained in execution:

```

PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-zyhajtvj.w5a' '--stdout=Microsoft-MIEngine-Out-t54f3212.4su' '--stderr=Microsoft-MIEngine-Error-ralac5gf.k4o' '--pid=Microsoft-MIEngine-Pid-gmu0ig4s.q4y' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'

Enter the number of rows: 6
1 2 3 4 5 6
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> []

```

4. (c) Display Pattern C

```
1  
2 1  
3 2 1  
4 3 2 1  
5 4 3 2 1  
6 5 4 3 2 1
```

**Answer:**

**Following code for input:**

```
/* read values from input file  
Practical 3, Part 4 - 3.1, Display Pattern C  
@Nirvik K.C. */  
  
#include <stdio.h>  
  
int main()  
{  
    int rows;  
    int i, j, k;  
    printf("Enter the number of rows: ");  
    scanf("%d", &rows);
```

```
for (int i = 1; i <= rows; i++)  
{  
    for (int j = 1; j <= (rows - i); j++) // calculating spaces  
    {  
        printf(" "); // printing two spaces  
    }  
    // Print descending numbers from i to 1  
    for (int k = i; k >= 1; k--)  
    {  
        printf("%d ", k);  
    }  
    printf("\n"); // go to next line after printing each row  
}  
return 0;  
}
```

### **Output obtained in execution:**

```
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS ⇧ cppdbg:DisplayPatternExe + v ⌂ ⌂ ... | ⌂ X ✓  
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '-stdin=Microsoft-MIEngine-In-kch3dhaj.fso' '-stdout=Microsoft-MIEngine-Out-yibr1lmy.ga2' '-stderr=Microsoft-MIEngine-Error-bzxhdqg.s3t' '--pid=Microsoft-MIEngine-Pid-0001kt5u.vqx' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'  
Enter the number of rows: 6  
1  
2 1  
3 2 1  
4 3 2 1  
5 4 3 2 1  
6 5 4 3 2 1  
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> [
```

#### 4. (d) Display Pattern D

1 2 3 4 5 6

1 2 3 4 5

1 2 3 4

1 2 3

12

1

**Answer:**

## Following code for input:

/\* read values from input file

Practical 3, Part 4 - 3.1, Display Pattern D

@Nirvik K.C. \*/

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int rows;
```

```
    printf("Enter the number of rows: ");
```

```
    scanf("%d", &rows);
```

```
    for (int i = rows; i >= 1; i--) // for number of rows
```

```
{
```

```
    for (int j = 1; j <= (rows - i); j++) // calculating spaces
```

```
{
```

```
    printf(" "); // printing two spaces
```

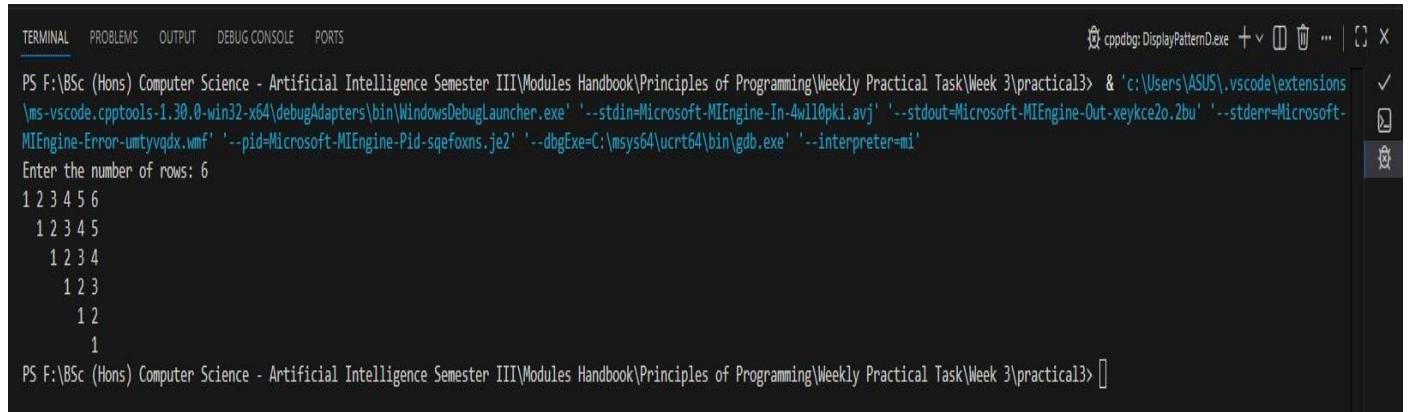
```
}
```

```
// Print descending numbers from i to 1
```

25024649

```
for (int k = 1; k <= i; k++)  
{  
    printf("%d ", k);  
  
}  
  
printf("\n"); // go to next line after printing each row  
  
}  
  
return 0;  
}
```

## Output obtained in execution:



The screenshot shows a terminal window in a dark-themed IDE interface. The terminal tab is selected at the top. The command entered is 'cppdbg:DisplayPatternD.exe'. The output shows the following pattern being printed:

```
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-4wll0pki.avj' '--stdout=Microsoft-MIEngine-Out-xeykce2o.2bu' '--stderr=Microsoft-MIEngine-Error-umtyvwdx.wmf' '--pid=Microsoft-MIEngine-Pid-sqefoxns.je2' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'  
Enter the number of rows: 6  
1 2 3 4 5 6  
1 2 3 4 5  
1 2 3 4  
1 2 3  
1 2  
1
```

5. (Occurrence of max numbers) Write a program that reads integers, finds the largest of them, and counts its occurrences. Assume that the input ends with number 0. Suppose that you entered 3 5 2 5 5 5 0; the program finds that the largest is 5 and the occurrence count for 5 is 4.

**Answer:****Following code for input:**

```
/* read values from input file
```

Practical 3, Part 4 - 3.2 (Occurrence of max numbers)

@Nirvik K.C. \*/

```
#include <stdio.h>
```

```
int main()
```

```
{
```

```
    int num;
```

```
    int number;
```

```
    int max;
```

```
    int count = 0;
```

```
printf("Enter numbers (type 0 to stop):\n");

// Read the first number

scanf("%d", &number);

while (number != 0)

{

    if (number > max)

    {

        max = number;

        count = 1; // starting a new count

    }

    else if (number == max)

    {

        count++;

    }

}
```

```
    scanf("%d", &number); // Read the next number

}

// Display the maximum number and its occurrence after 0 is entered at last

if (max == 0)

{

    printf("No numbers were entered.\n");

}

else

{

    printf("The largest number is %d.\n", max);

    printf("The occurrence of the largest number is %d times.\n", max, count);

}

return 0;
```

/\* Output:

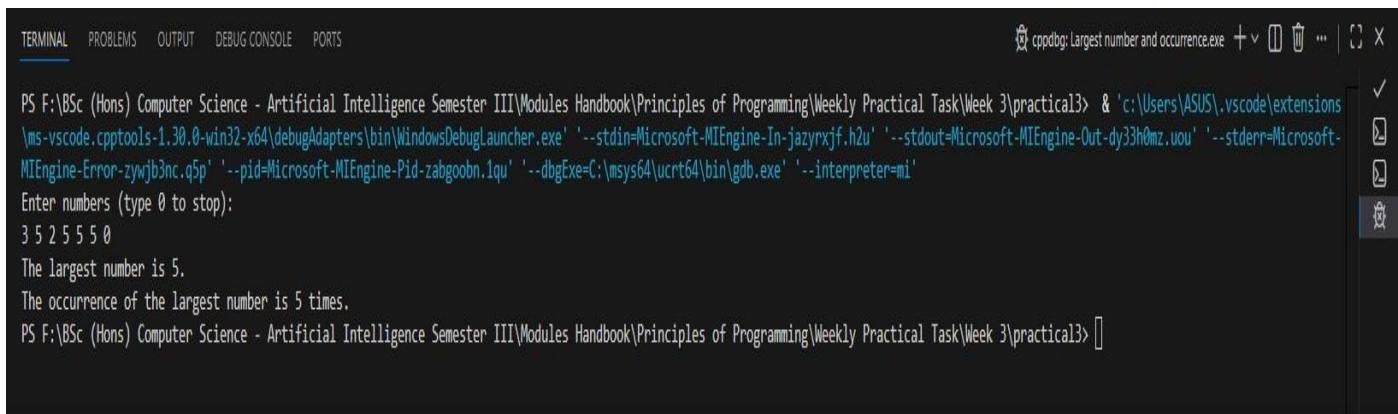
Enter numbers (type 0 to stop):

3 5 2 5 5 5 0

The largest number is 5.

The occurrence of the largest number is 5 times.\*/

## Output obtained in execution:



```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg: Largest number and occurrence.exe + v ⚡ ... | X

PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-jazyrxf.h2u' '--stdout=Microsoft-MIEngine-Out-dy33h0mz.uou' '--stderr=Microsoft-MIEngine-Error-zywjb3nc.q5p' '--pid=Microsoft-MIEngine-Pid-zabgoobn.1qu' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
Enter numbers (type 0 to stop):
3 5 2 5 5 5 0
The largest number is 5.
The occurrence of the largest number is 5 times.
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3>

```

6. (Count vowels and consonants) Assume letters A, E, I, O, and U as the vowels. Write a program that prompts the user to enter a string and displays the number of vowels and consonants in the string.

Output to display:

Enter a string: Programming is fun

The number of vowels is 5

The number of consonants is 11

**Answer:**

**Following code for input:**

```
/* read values from input file
```

Practical 3, Part 4 - 3.3 (Count vowels and consonants)

@Nirvik K.C. \*/

```
#include <stdio.h>

#include <string.h>

int main()

{

char str[100];

int vowels = 0, consonants = 0;

int i, j;

char vowel_list[10] = {'A', 'E', 'I', 'O', 'U', 'a', 'e', 'i', 'o', 'u'};

printf("\nEnter a string: ");

// To read the whole line including spaces

fscanf(stdin, "%[^\\n]", str); // using %[^\\n] to read string with spaces

for (int i = 0; i < strlen(str); i++)

{



char ch = str[i];
```

```
// Check if the character is a letter (A-Z or a-z)

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z'))

{

    int isVowel = 0;

    for (int j = 0; j < 10; j++)

    {

        if (ch == vowel_list[j])

        {

            isVowel = 1;

            break;

        }

    }

    if (isVowel)

    {

        vowels = vowels + 1;

    }

}
```

```
else
{
    consonants = consonants + 1;

}
}

printf("The number of vowels is %d\n", vowels);
printf("The number of consonants is %d\n", consonants);
return 0;
}
```

/\* Output:

Enter a string: Programming is fun

The number of vowels is 5

The number of consonants is 11 \*/

## Output obtained in execution:

The screenshot shows a terminal window with the following content:

```
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE PORTS
cppdbg: Displays the number of vowels and consonants in the string.exe + v ⌂ ⌄ ... | ↴
```

```
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> & 'c:\Users\ASUS\.vscode\extensions\ms-vscode.cpptools-1.30.0-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-dguqohzw.csj' '--stdout=Microsoft-MIEngine-Out-rfih4mn3.f34' '--stderr=Microsoft-MIEngine-Error-3q0rhopa.41q' '--pid=Microsoft-MIEngine-Pid-iqdtpou.gbg' '--dbgExe=C:\msys64\ucrt64\bin\gdb.exe' '--interpreter=mi'
```

```
Enter a string: Programming is fun
The number of vowels is 5
The number of consonants is 11
PS F:\BSc (Hons) Computer Science - Artificial Intelligence Semester III\Modules Handbook\Principles of Programming\Weekly Practical Task\Week 3\practical3> 
```