

PPS1

Q1

Aim:

Write a program in C to find the roots of a quadratic equation.

Procedure:

Input:

Coefficients a, b and c

Output:

Roots

Algorithm:

Step 1: Read a, b and c

Step 2: Discriminant, $d = b^2 - 4ac$

Step 3: If discriminant is greater than 0

Then roots are real and distinct

Root1 = $(-b + \sqrt{d})/2a$, Root2 = $(-b - \sqrt{d})/2a$

Step 4: If discriminant is equal to 0

Then roots are real and equal

Root1 = Root2 = $-b/2a$

Step 3: If discriminant is less than 0

Then roots are imaginary

Real Part = $-b/2a$, Imaginary Part = $\sqrt{-d}/2a$

Root1 = realPart + imagPart(i), Root2 = realPart – imagPart(i)

Step 6: Display the roots

Code:

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

int main() {
    double a, b, c, d, r1, r2, realPart, imagPart;
    printf("\nCoefficients a, b and c are: ");
    scanf("%lf %lf %lf", &a, &b, &c);

    d = pow(b,2) - 4 * a * c;
```

```

// Real and distinct roots
if (d > 0) {
    r1 = (-b + sqrt(d)) / (2 * a);
    r2 = (-b - sqrt(d)) / (2 * a);
    printf("\nQuadratic Equation has real and distinct roots:\n");
    printf("Root 1 = %.2lf and Root 2 = %.2lf\n\n", r1, r2);
}

// Real and equal roots
else if (d == 0) {
    r1 = r2 = -b / (2 * a);
    printf("\nQuadratic Equation has real and equal roots:\n");
    printf("Root 1 = Root 2 = %.2lf\n\n", r1);
}

// Imaginary roots
else {
    realPart = -b / (2 * a);
    imagPart = sqrt(-d) / (2 * a);
    printf("\nQuadratic Equation has imaginary roots:\n");
    printf("Root 1 = %.2lf + %.2lfi and Root 2 = %.2f - %.2fi\n\n",
        realPart, imagPart, realPart, imagPart);
}

return 0;
}

```

The screenshot shows a Visual Studio Code editor with a C++ file named 'qeroots.c'. The code implements a program to solve quadratic equations based on the discriminant (d). The program is being compiled and run in a terminal window. The output shows three cases: real and distinct roots, real and equal roots, and imaginary roots.

```

C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> gcc -o qeroots qeroots.c
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> ./qeroots
Coefficients a, b and c are: 1 -3 2
Quadratic Equation has real and distinct roots:
Root 1 = 2.00 and Root 2 = 1.00
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> gcc -o qeroots qeroots.c
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> ./qeroots
Coefficients a, b and c are: 1 -2 1
Quadratic Equation has real and equal roots:
Root 1 = Root 2 = 1.00
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> gcc -o qeroots qeroots.c
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1> ./qeroots
Coefficients a, b and c are: 1 -2 2
Quadratic Equation has imaginary roots:
Root 1 = 1.00 + 1.00i and Root 2 = 1.00 - 1.00i
PS C:\Users\vidhi\Shah\Desktop\VIITC Sem 2\C++ Lab\PPS1>

```

Q2

Aim:

Write a program in C to create a simple calculator.

Procedure:

Input:

Operator, op

Operands, n1 and n2

Output:

Equation with answer

Algorithm:

Step 1: Read operator and operands, op, n1, n2

Step 2: Use switch case for operator

Case 1 ('+'): Result = $n1 + n2$

Case 2 ('-'): Result = $n1 - n2$

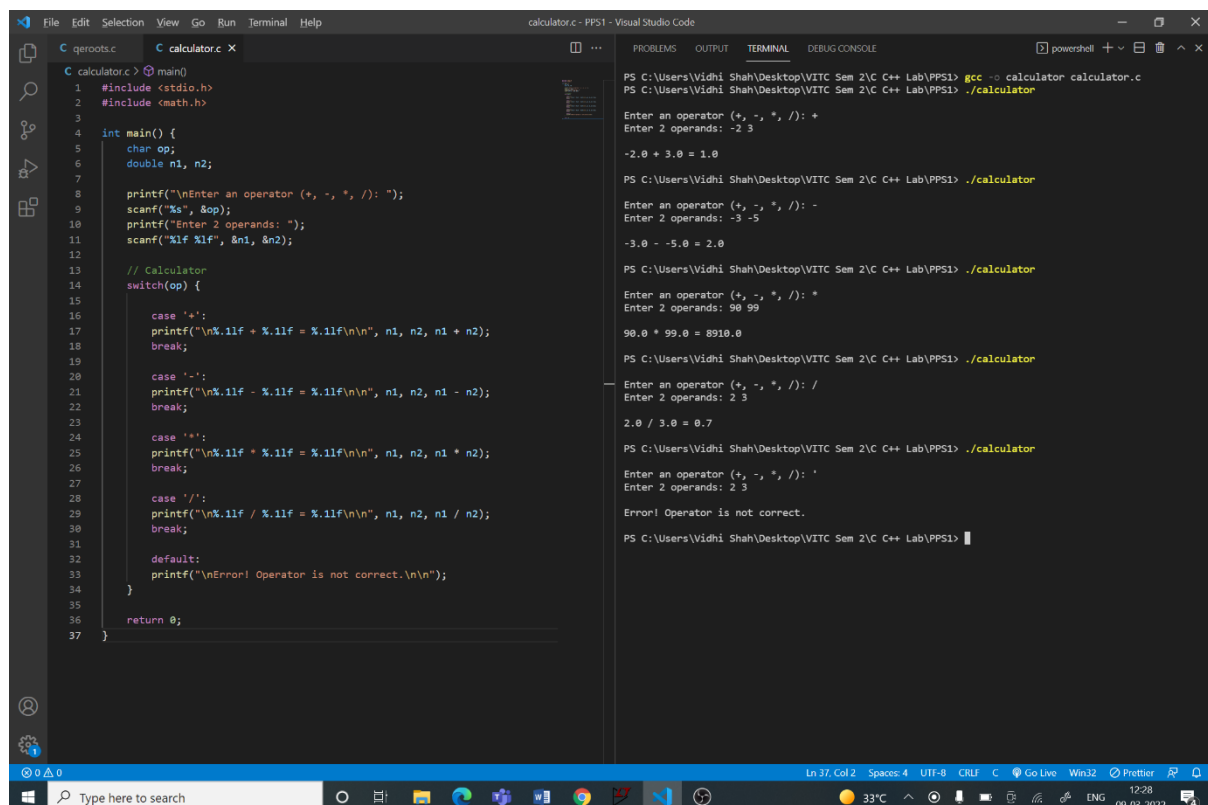
Case 3 ('*'): Result = $n1 * n2$

Case 4 ('/'): Result = $n1 / n2$

Default: Error message for invalid input

Step 3: Print the result

Code:



```
calculator.c - PPS1 - Visual Studio Code
C qeroots.c C calculator.c X
C calculator.c (main)
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     char op;
6     double n1, n2;
7
8     printf("\nEnter an operator (+, -, *, /): ");
9     scanf("%s", &op);
10    printf("Enter 2 operands: ");
11    scanf("%lf %lf", &n1, &n2);
12
13    // Calculator
14    switch(op) {
15        case '+':
16            printf("\n%.1lf + %.1lf = %.1lf\n", n1, n2, n1 + n2);
17            break;
18        case '-':
19            printf("\n%.1lf - %.1lf = %.1lf\n", n1, n2, n1 - n2);
20            break;
21        case '*':
22            printf("\n%.1lf * %.1lf = %.1lf\n", n1, n2, n1 * n2);
23            break;
24        case '/':
25            printf("\n%.1lf / %.1lf = %.1lf\n", n1, n2, n1 / n2);
26            break;
27        default:
28            printf("\nError! Operator is not correct.\n");
29    }
30
31    return 0;
32 }
```

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> gcc -o calculator calculator.c

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> ./calculator

Enter an operator (+, -, *, /): +

Enter 2 operands: -2 3

-2.0 + 3.0 = 1.0

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> ./calculator

Enter an operator (+, -, *, /): -

Enter 2 operands: -3 -5

-3.0 - -5.0 = 2.0

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> ./calculator

Enter an operator (+, -, *, /): *

Enter 2 operands: 90 99

90.0 * 99.0 = 8910.0

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> ./calculator

Enter an operator (+, -, *, /): /

Enter 2 operands: 2 3

2.0 / 3.0 = 0.7

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> ./calculator

Enter an operator (+, -, *, /): '

Enter 2 operands: 2 3

Error! Operator is not correct.

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS1> |

Q3

Aim:

Write a program in C to display the days of the week getting the user input as a character indicating the first letter of the day. Demonstrate the use of Switch case statements.

Procedure:

Input:

Character indicating the first letter of the day, wd

Output:

Day

Algorithm:

Step 1: Read character, wd

Step 2: Use switch case for character

Case 1 ('M', 'm'): Day = Monday

Case 2 ('T'): Day = Tuesday

Case 3 ('W', 'w'): Day = Wednesday

Case 4 ('t'): Day = Thursday

Case 5 ('F', 'f'): Day = Friday

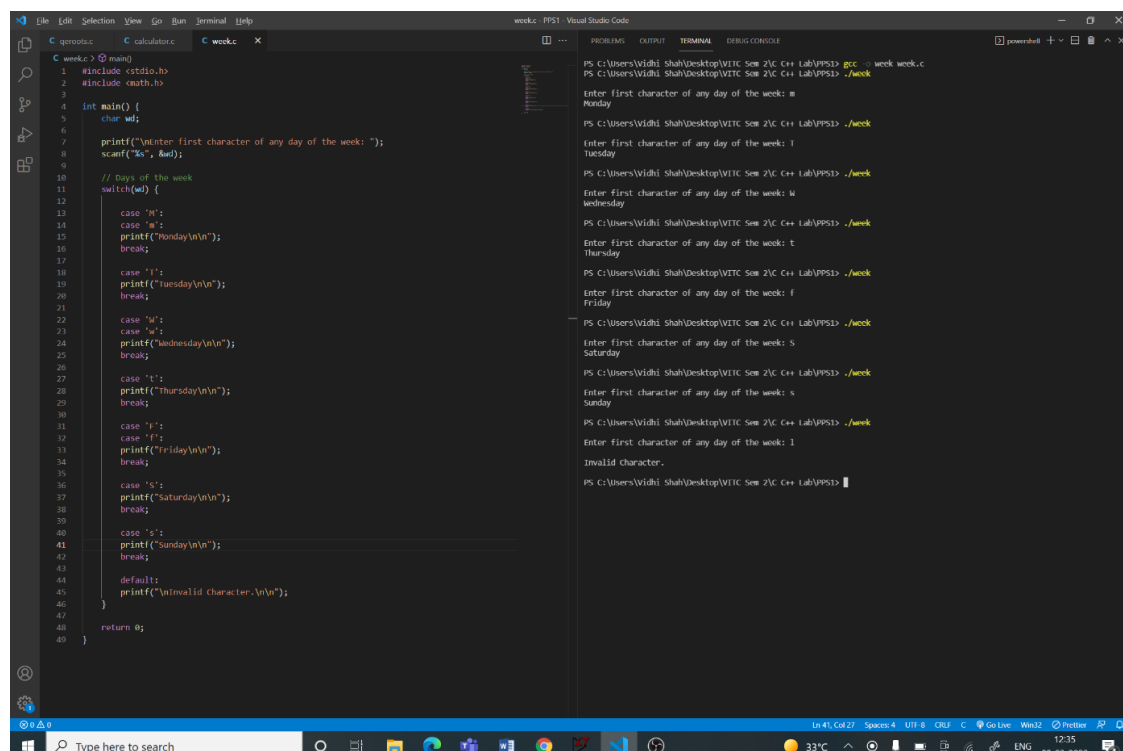
Case 6 ('S'): Day = Saturday

Case 7 ('s'): Day = Sunday

Default: Error message for invalid input

Step 6: Print the day

Code:



```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     char wd;
6
7     printf("\nEnter first character of any day of the week: ");
8     scanf("%s", &wd);
9
10    // Days of the week
11    switch(wd) {
12
13        case 'M':
14        case 'm':
15            printf("Monday\n");
16            break;
17
18        case 'T':
19            printf("Tuesday\n");
20            break;
21
22        case 'W':
23        case 'w':
24            printf("Wednesday\n");
25            break;
26
27        case 't':
28            printf("Thursday\n");
29            break;
30
31        case 'F':
32            printf("Friday\n");
33            break;
34
35        case 'S':
36            printf("Saturday\n");
37            break;
38
39        case 's':
40            printf("Sunday\n");
41            break;
42
43        default:
44            printf("Invalid Character.\n");
45    }
46
47    return 0;
48 }
```

Terminal Output:

```
PS C:\Users\Widhi> gcc -o week week.c
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: m
Monday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: T
Tuesday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: W
Wednesday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: t
Thursday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: f
Friday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: S
Saturday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: s
Sunday
PS C:\Users\Widhi> ./week
Enter first character of any day of the week: l
Invalid Character.
PS C:\Users\Widhi>
```

PPS2

Q1

Aim:

Write a program in C using while loop structure to display the sum of first n natural numbers.

Procedure:

Input:

Natural number, n

Output:

Sum of n natural numbers

Algorithm:

Step 1: Read n, initialize i to 1 and sum to 0

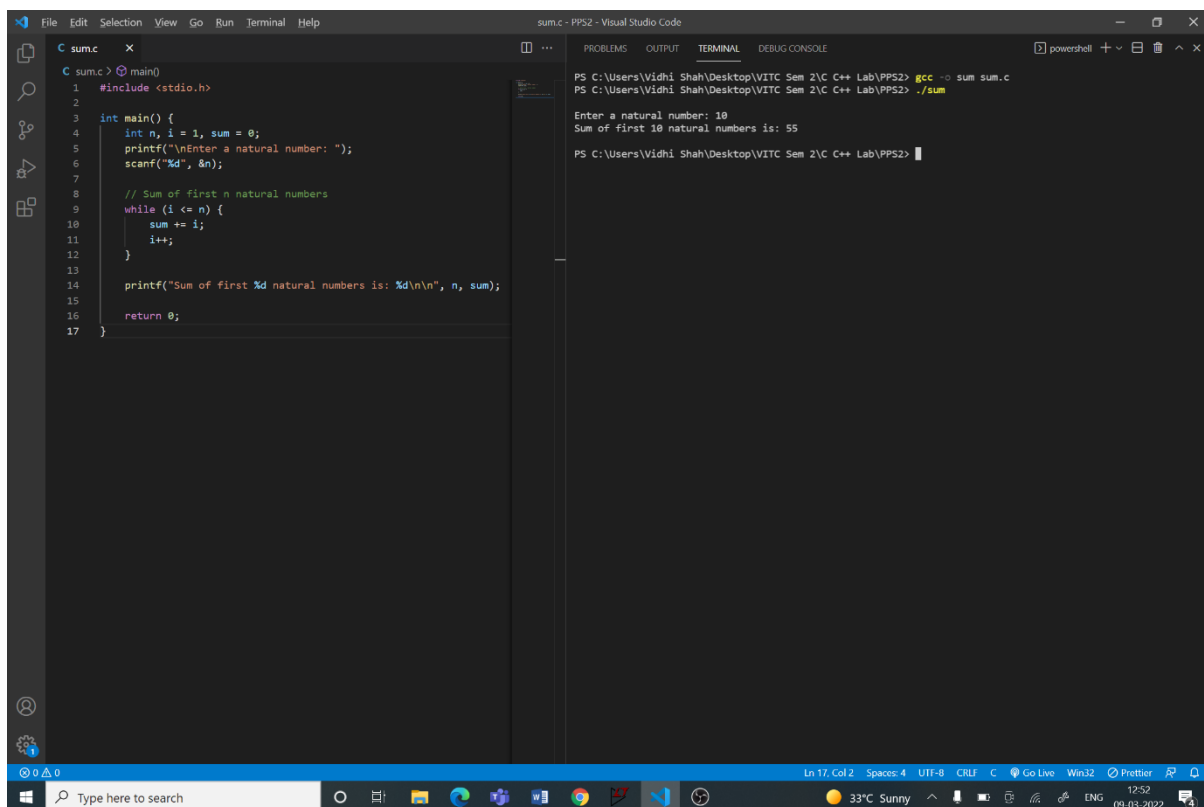
Step 2: Using while loop until i is less than equal to n

 Add i to sum

 Increment i by 1

Step 3: Print the sum

Code:



```
sum.c - PPS2 - Visual Studio Code

C sum.c x
1 #include <stdio.h>
2
3 int main() {
4     int n, i = 1, sum = 0;
5     printf("\nEnter a natural number: ");
6     scanf("%d", &n);
7
8     // Sum of first n natural numbers
9     while (i <= n) {
10        sum += i;
11        i++;
12    }
13
14    printf("Sum of first %d natural numbers is: %d\n", n, sum);
15
16    return 0;
17 }
```

```
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS2> gcc -o sum sum.c
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS2> ./sum
Enter a natural number: 10
Sum of first 10 natural numbers is: 55
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C C++ Lab\PPS2>
```

Q2

Aim:

Write a program in C using for loop structure to find the sum and average of n numbers.

Procedure:

Input:

Number of elements, n

Next n lines contain n numbers

Output:

Sum of the numbers

Average of the numbers

Algorithm:

Step 1: Read n, initialise sum to 0

Step 2: Using for loop initialise i to one. Until i is less than or equal to n

 Read a number

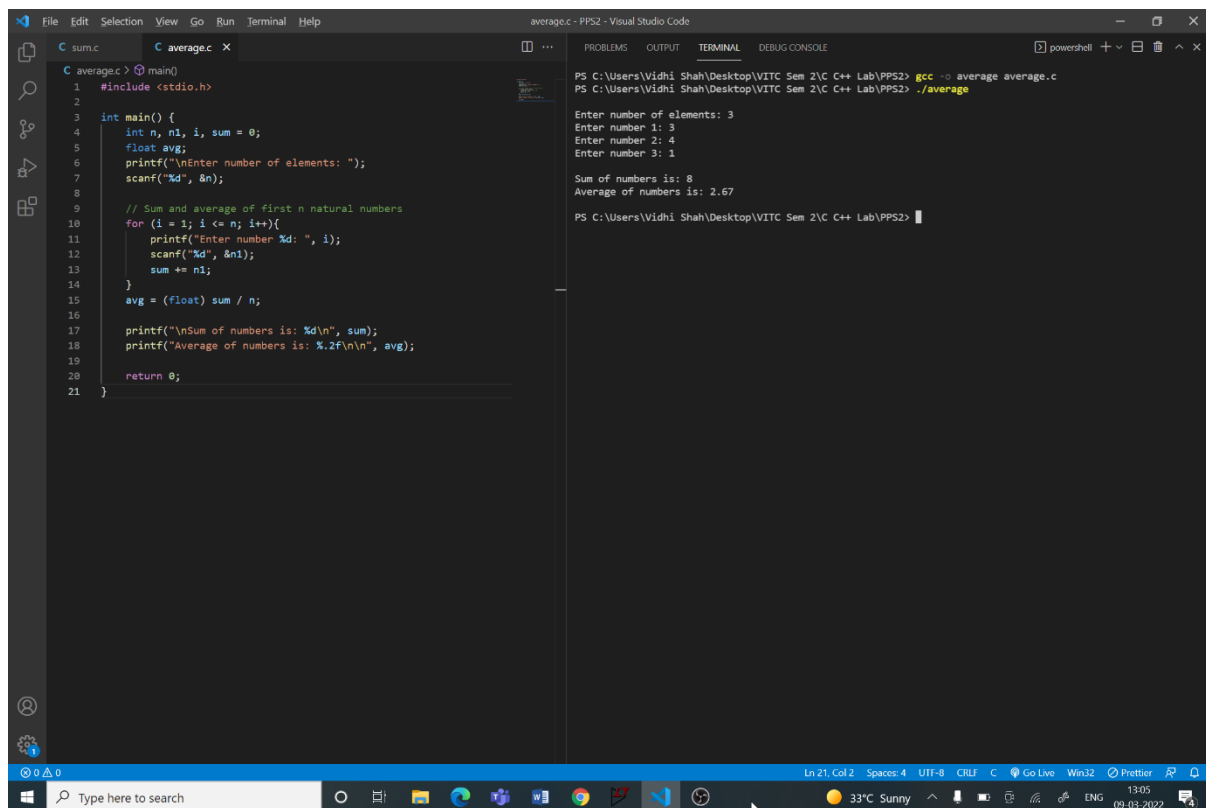
 Add the number to sum

 Increment i by 1

Step 3: Average = Sum / Number of elements

Step 4: Print sum and average of the numbers in separate line

Code:



```
average.c - PPS2 - Visual Studio Code
File Edit Selection View Go Run Terminal Help
C average.c x
1 #include <stdio.h>
2
3 int main() {
4     int n, n1, i, sum = 0;
5     float avg;
6     printf("\nEnter number of elements: ");
7     scanf("%d", &n);
8
9     // Sum and average of first n natural numbers
10    for (i = 1; i <= n; i++){
11        printf("Enter number %d: ", i);
12        scanf("%d", &n1);
13        sum += n1;
14    }
15    avg = (float) sum / n;
16
17    printf("\nSum of numbers is: %d\n", sum);
18    printf("Average of numbers is: %.2f\n", avg);
19
20    return 0;
21 }

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPS2> gcc -o average average.c
PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPS2> ./average
Enter number of elements: 3
Enter number 1: 3
Enter number 2: 4
Enter number 3: 1
Sum of numbers is: 8
Average of numbers is: 2.67
PS C:\Users\Vidhi Shah\Desktop\VIITC Sem 2\C++ Lab\PPS2>

Ln 21, Col 2 Spaces: 4 UTF-8 CRLF C Go Live Win32 Prettier
```

Q3

Aim:

Write a program in C using for loops to display the pattern like right angle triangle using an asterisk.

Procedure:

Input:

Height of the triangle, h

Output:

Right angled triangle of height 'h' using asterisks

Algorithm:

Step 1: Read h

Step 2: Use for loop until i is less than h. Initialise i to 1. For each iteration:

Step A: Use for loop until j is less than equal to i. Initialise j to 0.

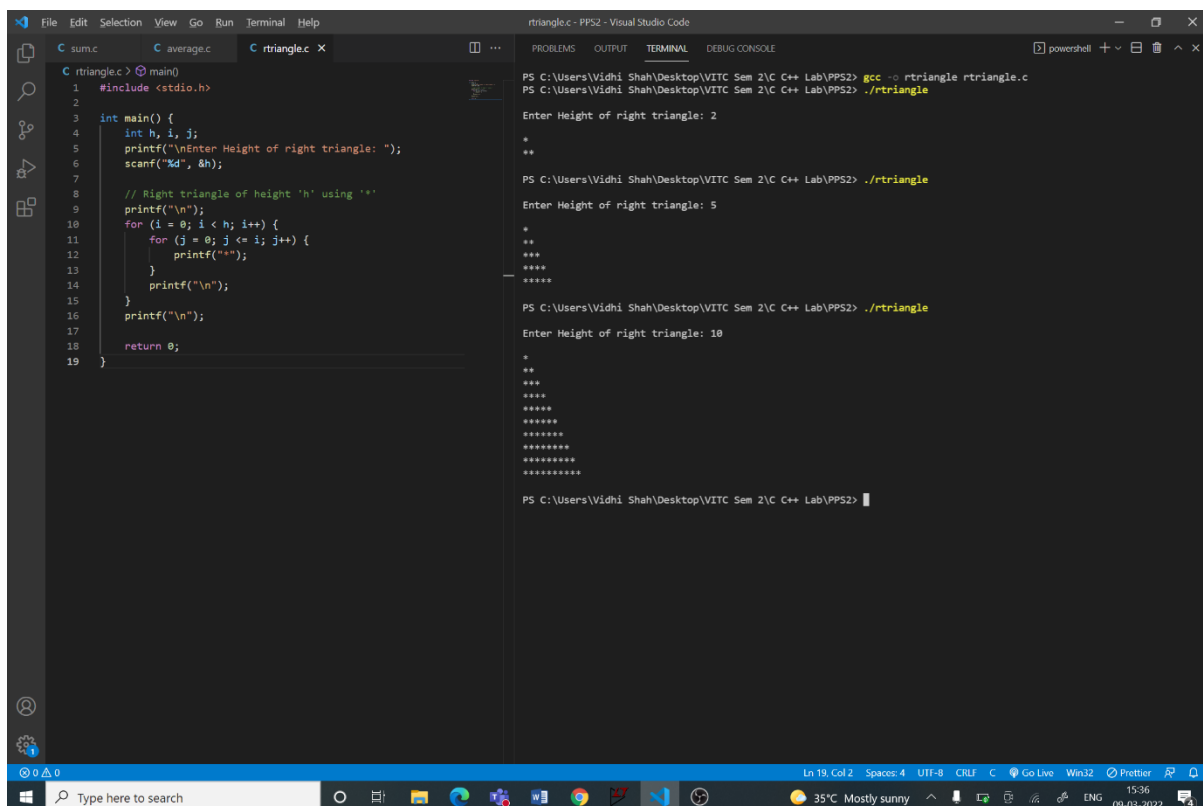
Step i: Print '*' symbol

Step ii: Increment j

Step B: Print a new line

Step C: Increment i

Code:



```
1 #include <stdio.h>
2
3 int main() {
4     int h, i, j;
5     printf("\nEnter Height of right triangle: ");
6     scanf("%d", &h);
7
8     // Right triangle of height 'h' using '*'
9     printf("\n");
10    for (i = 0; i < h; i++) {
11        for (j = 0; j <= i; j++) {
12            printf("*");
13        }
14        printf("\n");
15    }
16    printf("\n");
17    return 0;
18 }
```

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS2> gcc -o rtriangle rtriangle.c

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS2> ./rtriangle

Enter Height of right triangle: 2

```
*
**
```

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS2> ./rtriangle

Enter Height of right triangle: 5

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

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS2> ./rtriangle

Enter Height of right triangle: 10

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

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS2>

PPS3

Q1

Aim:

Write a program in C to print the pattern:

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

Procedure:

Input:

No input

Output:

Above Pattern

Algorithm:

Step 1: Use for loop until i is less than 10. Initialise i to 1. For each iteration:

Step A: Use for loop until j is less than equal to i. Initialise j to 0.

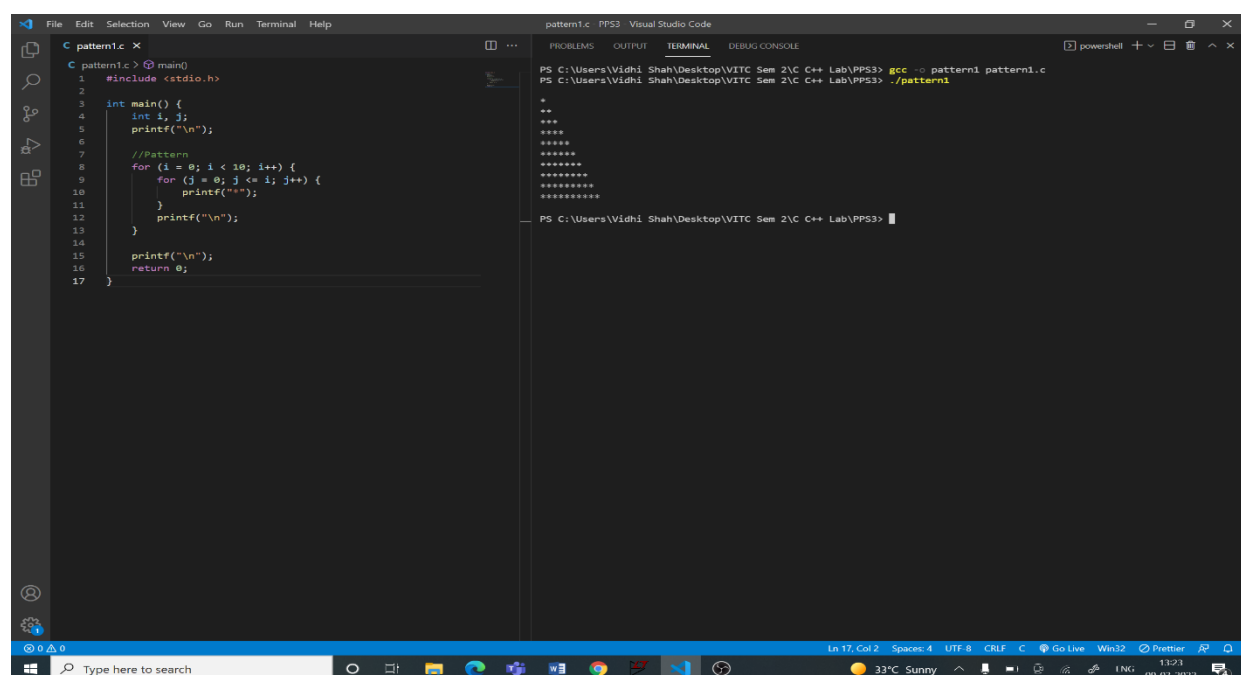
Step i: Print '*' symbol

Step ii: Increment j

Step B: Print a new line

Step C: Increment i

Code:



The screenshot displays the Visual Studio Code editor with a C program named `pattern1.c` open. The code is as follows:

```
1 #include <stdio.h>
2
3 int main() {
4     int i, j;
5     printf("\n");
6
7     //Pattern
8     for (i = 0; i < 10; i++) {
9         for (j = 0; j <= i; j++) {
10             printf("*");
11         }
12         printf("\n");
13     }
14
15     printf("\n");
16     return 0;
17 }
```

The terminal window on the right shows the command `gcc -o pattern1 pattern1.c` and `./pattern1` being executed, resulting in the pattern of asterisks shown in the output section above.

Q2

Aim:

Write a program in C to print the pattern:

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

Procedure:

Input:

No input

Output:

Above Pattern

Algorithm:

Step 1: Use for loop until i is greater than 0. Initialise i to 1. For each iteration:

Step A: Use for loop until j is less than i. Initialise j to 0.

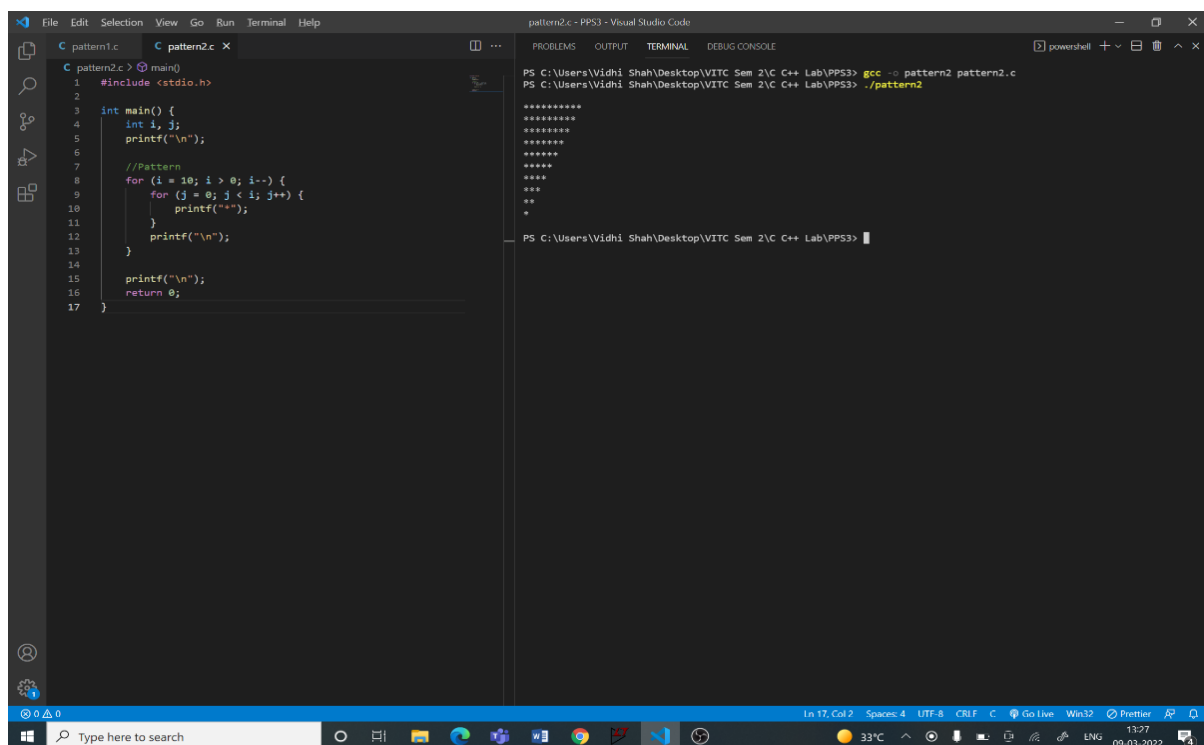
Step i: Print '*' symbol

Step ii: Increment j

Step B: Print a new line

Step C: Increment i

Code:



The screenshot shows the Visual Studio Code editor with a C program in a file named `pattern2.c`. The code is as follows:

```
1 #include <stdio.h>
2
3 int main() {
4     int i, j;
5     printf("\n");
6
7     //Pattern
8     for (i = 10; i > 0; i--) {
9         for (j = 0; j < i; j++) {
10             printf("*");
11         }
12         printf("\n");
13     }
14
15     printf("\n");
16     return 0;
17 }
```

The terminal window shows the compilation and execution of the program:

```
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS3> gcc -o pattern2 pattern2.c
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS3> ./pattern2
*****
*****
*****
*****
*****
****
***
**
*
```

The status bar at the bottom indicates the file is at line 17, column 2, with 4 spaces, UTF-8 encoding, CRLF line endings, and C++ language. The system tray shows a temperature of 33°C and the date 09-03-2022.

Q3

Aim:

Write a program in C to print the pattern:

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

Procedure:

Input:

No input

Output:

Above Pattern

Code:

```
#include <stdio.h>

int main() {
    int i, j, n = 9;
    printf("\n");

    //Upper Triangle
    for (i = 1; i <= n; i = i + 2) {
        for (j = 0; j < (n-i)/2; j++) {
            printf(" ");
        }

        for (j = 0; j < i; j++) {
            printf("*");
        }

        for (j = 0; j < (n-i)/2; j++) {
            printf(" ");
        }

        printf("\n");
    }

    //Lower Triangle
    for (i = n - 2; i > 0; i = i - 2) {
        for (j = 0; j < (n-i)/2; j++) {
            printf(" ");
        }
    }
}
```

```

        for (j = 0; j < i; j++) {
            printf("*");
        }

        for (j = 0; j < (n-i)/2; j++) {
            printf(" ");
        }

        printf("\n");

    }

    printf("\n");
    return 0;
}

```

The screenshot shows the Visual Studio Code editor with a C++ file named `pattern3.c`. The code implements a function `main()` that prints a diamond pattern. The pattern consists of an upper triangle of asterisks, a middle row of spaces, and a lower triangle of asterisks. The terminal output shows the pattern for `n=5`.

```

1 #include <stdio.h>
2
3 int main() {
4     int i, j, n = 0;
5     printf("\n");
6
7     //Upper Triangle
8     for (i = 1; i <= n; i = i + 2) {
9         for (j = 0; j < (n-i)/2; j++) {
10             printf(" ");
11         }
12
13         for (j = 0; j < i; j++) {
14             printf("*");
15         }
16
17         for (j = 0; j < (n-i)/2; j++) {
18             printf(" ");
19         }
20
21         printf("\n");
22     }
23
24     //Lower Triangle
25     for (i = n - 2; i > 0; i = i - 2) {
26         for (j = 0; j < (n-i)/2; j++) {
27             printf(" ");
28         }
29
30         for (j = 0; j < i; j++) {
31             printf("*");
32         }
33
34         for (j = 0; j < (n-i)/2; j++) {
35             printf(" ");
36         }
37
38         printf("\n");
39     }
40
41     printf("\n");
42     return 0;
43 }

```

The terminal output shows the pattern for `n=5`:

```

PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS3> gcc -o pattern3 pattern3.c
PS C:\Users\Vidhi Shah\Desktop\VITC Sem 2\C++ Lab\PPS3> ./pattern3
*
***
*****
*****
*****
***
*

```