VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

B.Tech Programme: AI&ML
Course Title: Programming in C
Course Code: ES-101

Submitted to:
Prof Vikas Goel

Submitted By:
Name: **Bhavesh Upadhyay**
Enrolment No:

VIPS
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

VISION OF INSTITUTE

To be an educational institute that empowers the field of engineering to build a sustainable future by providing quality education with innovative practices that supports people, planet and profit.

MISSION OF INSTITUTE

To groom the future engineers by providing value-based education and awakening students' curiosity, nurturing creativity and building capabilities to enable them to make significant contributions to the world.

VIPS
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

INDEX (GGSIPU)

S.No	Experiment	Date	Marks			Remark	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								

VIPS
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

INDEX (BEYOND CURRICULUM)

S.No	Experiment	Date	Marks			Remark	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								

VIPS
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC

NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

INDEX (PRACTICE)

S.No	Experiment	Date	Marks			Remark	Updated Marks	Faculty Signature
			Laboratory Assessment (15 Marks)	Class Participation (5 Marks)	Viva (5 Marks)			
1								
2								
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

GGSIPU EXPERIMENTS

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

BEYOND CURRICULAM EXPERIMENTS

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

PRACTICE EXPERIMENTS

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

Average of numbers.

PROBLEM STATEMENT:

Write a C program to find average of 3 numbers.

C PROGRAM:

```c
// Write a C program to find average of 3 numbers.

#include <stdio.h>

int main()
{
    // Declaration
    int num1, num2, num3;
    float avg;

    // Reading Input
    printf("Enter 3 numbers with a blankspace in between: ");
    scanf("%d %d %d", &num1, &num2, &num3);

    // Computing Average
    avg = (float)(num1 + num2 + num3)/3;
    printf("Average of 3 numbers is %f", avg);

    return 0;
}
```

OUTPUT:

```
Enter 3 numbers with a blankspace in between: 23 11 6
Average of 3 numbers is 13.333333
```

LEARNING OUTCOMES:

The program demonstrates correct variable declaration using appropriate data types.

It utilizes the scanf function for efficient user input of multiple values in a single statement.

Type casting to float is employed to ensure accurate calculation of the average, avoiding truncation from integer division.

The output is displayed using the printf function with the %f format specifier, which is suitable for floating-point numbers.

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

Area and Perimeter of circle

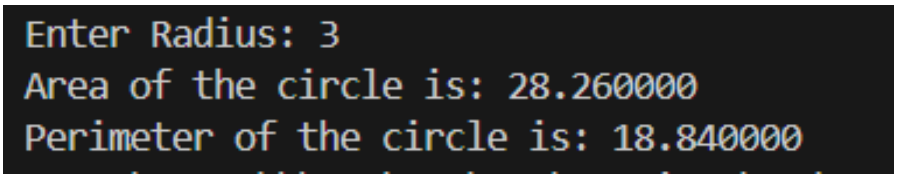
PROBLEM STATEMENT:

Write a C program to calculate and print area and perimeter, taking its radius as input .

C PROGRAM:

```c
/*
    Write a C program to calculate and print
    area and perimeter of a circle, taking its radius as input.
*/
#include <stdio.h>
int main()
{
    //Declaration
    float PI = 3.14, radius, area, perimeter;

    // Reading Input
    printf("Enter Radius: ");
    scanf("%f", &radius);

    // Computing Area
    area = (PI*radius*radius);

    // Computing Perimeter
    perimeter = (2*PI*radius);

    // Printing Output
    printf("Area of the circle is: %f\n", area);
    printf("Perimeter of the circle is: %f\n", perimeter);
    return 0;
}
```

OUTPUT:



LEARNING OUTCOMES:

The program illustrates variable declaration with the use of float to accommodate decimal values for radius, area, and perimeter.

It employs the constant variable PI, facilitating calculation based on geometric formulas.

User input is captured through scanf, allowing dynamic calculation based on the input radius.

Output formatting uses printf and the %f specifier for floating-point presentation, ensuring accurate results are displayed to the user.

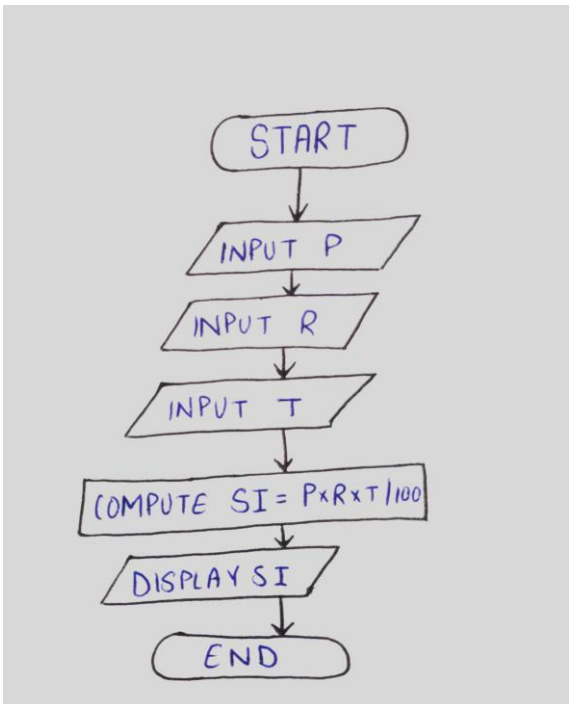
EXPERIMENT:

Calculate Simple Interest.

PROBLEM STATEMENT:

Write a C program to find simple interest. Given principal amount, rate of interest per year and tenure (in years). Draw flowchart and write an algorithm for the same

FLOWCHART:



ALGORITHM:

Step 0 : START
Step 1 : INPUT principal amount into variable P
Step 2 : INPUT rate of interest into variable R
Step 3 : INPUT duration into variable T
Step 4 : COMPUTE Simple_Interest = P×R×T/100
Step 5 : DISPLAY Simple_Interest
Step 6 : END

C PROGRAM:

```c
/*
    Write a C progam to find simple interest. given principal amount,
    rate of interest per year and tenure (in years)
*/
#include <stdio.h>
int main()
{
    //Declaration
    float SI, principal_amt, rate, time;

    printf("Enter Principal Amount: ");
    scanf("%f", &principal_amt);

    printf("Enter rate of interest: ");
    scanf("%f", &rate);

    printf("Enter tenure: ");
    scanf("%f", &time);
    // Computing Simple Interese
    SI = (principal_amt * rate * time)/100;

    // Printing Output
    printf("Simple Interest: %f\n", SI);
    return 0;
}
```

OUTPUT:

```
Enter Principal Amount: 60000
Enter rate of interest: 8.5
Enter tenure: 3
Simple Interest: 15300.000000
```

LEARNING OUTCOMES:

The exercise illustrates proper use of the char data type for character input and manipulation in C.

The program demonstrates efficient input handling using scanf with the %c format specifier to read a single character from the user.

It reinforces understanding of ASCII encoding, showing how characters are internally represented by specific numeric values in computer systems.

The use of printf with the %d format specifier highlights the conversion of a character's symbolic form to its corresponding integer value, supporting formatted output practices.

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

Character to ASCII value

PROBLEM STATEMENT:

Write a C program to print ASCII value of a character takes as input

C PROGRAM:

```c
// Write a C program to print ASCII value of a character takes as input

#include <stdio.h>
int main()
{
    // Declaration
    char charVal;

    // Reading Input
    printf("Enter Character: ");
    scanf("%c", &charVal);

    // Printing ASCII value
    printf("ASCII value for given character is %d\n", charVal);
    return 0;
}
```

OUTPUT:

```
Enter Character: J
ASCII value for given character is 74
```

LEARNING OUTCOMES:

The exercise demonstrates stepwise conversion of an algorithm and flowchart into executable code, strengthening algorithmic thinking and programming design skills.

Input handling for principal, rate, and time variables reinforces user interaction concepts in C.

The direct implementation of the simple interest formula in code illustrates precise use of mathematical operators and operator precedence.

Output presentation with formatted printf statements enhances understanding of result communication in software applications.

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

ASCII value to character

PROBLEM STATEMENT:

Write a C program to print character when ASCII value is taken as input and also check that ASCII input is not greater than 127 (and if so, gives an error and terminates the program).

C PROGRAM:

```c
/*
    Write a C program to print character when ASCII value is taken
    as input and also check that ASCII input is not greater than 127
    (and if so, gives an error and terminates the program).
*/

#include <stdio.h>
int main()
{
    // Declaration
    int asciiVal;

    // Reading Input
    printf("Enter ASCII Value: ");
    scanf("%d", &asciiVal);

    // Printing ASCII value
    if((asciiVal > 127) || (asciiVal < 0))
    {
        printf("Error! Invalid ASCII value");
    }
    else
    {
        printf("Character for given ASCII Value is %c\n", asciiVal);
    }
    return 0;
}
```

OUTPUT:

```
Enter ASCII Value: -4
Error! Invalid ASCII value
```

```
Enter ASCII Value: 234
Error! Invalid ASCII value
```

```
Enter ASCII Value: 97
Character for given ASCII Value is a
```

LEARNING OUTCOMES:

It illustrates conditional logic through the use of an if-else statement, enabling the programmer to handle invalid data gracefully and provide appropriate error messaging to the user.

The code promotes understanding of character encoding, showing how integer values map to corresponding ASCII characters using the %c format specifier in output.

The exercise provides practical experience in structuring user interactions, validating inputs, and generating informative output within a C programming context

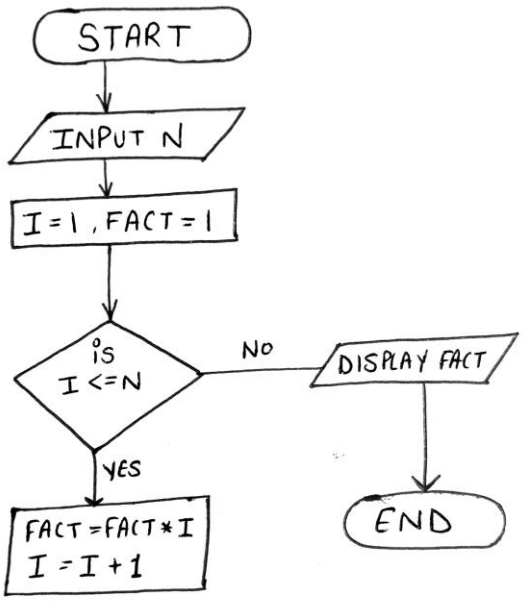
EXPERIMENT:

Factorial N

PROBLEM STATEMENT:

Write an algorithm and draw a flowchart for computing factorial N (N!) where N! = 1*2*3*4...*N.

FLOWCHART:



ALGORITHM:

Step 0 START
Step 1 INPUT N
Step 2 I = 1
Step 3 FACT = 1
Step 4 REPEAT Step (a) - (b) WHILE I <= N
(a) FACT = FACT * I
(b) I = I + 1
END WHILE
Step 5 DISPLAY FACT
Step 6 END

LEARNING OUTCOMES:

Looping and repetition in algorithms, shown by incrementing a counter until reaching the desired value.

Stepwise multiplication to calculate factorial, demonstrating use of arithmetic operations within a process.

Use of logical conditions to control how long a process continues.

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

Swap two variables

PROBLEM STATEMENT:

Write a C program to swap two variables with and without using 3rd variable.

LEARNING OUTCOME:

Each swap method (with and without a third variable) keeps its variables inside its own function, preventing mix-ups.

Builds an understanding of how value assignment works step by step, especially in the arithmetic method where each operation changes the state of both variables.

C PROGRAM:

```c
// Write a C program to swap two variables with and without using 3rd variable.

#include <stdio.h>

int twoVariables()
{
    printf("Swap 2 number without using 3rd variable\n");
    int num1, num2;
    printf("Enter 1st number: ");
    scanf("%d/n", &num1);
    printf("Enter 2nd number: ");
    scanf("%d/n", &num2);

    printf("1st Number and 2nd Number BEFORE Swap: %d %d\n", num1, num2);
    num1 = num1 + num2;
    num2 = num1 - num2;
    num1 = num1 - num2;
    printf("1st Number and 2nd Number AFTER Swap: %d %d\n", num1, num2);
}

int threeVariables()
{
    printf("Swap 2 number using 3rd variable\n");
    int num1, num2, temp;
    printf("Enter 1st number: ");
    scanf("%d/n", &num1);
    printf("Enter 2nd number: ");
    scanf("%d/n", &num2);
    printf("1st Number and 2nd Number BEFORE Swap: %d %d\n", num1, num2);

    temp = num1;
    num1 = num2;
    num2 = temp;

    printf("1st Number and 2nd Number AFTER Swap: %d %d\n", num1, num2);
}

int main(){
    threeVariables();
    twoVariables();
}
```

OUTPUT:

```
Swap 2 number using 3rd variable
Enter 1st number: 23
Enter 2nd number: 11
1st Number and 2nd Number BEFORE Swap: 23 11
1st Number and 2nd Number AFTER Swap: 11 23
Swap 2 number without using 3rd variable
Enter 1st number: 20
Enter 2nd number: 06
1st Number and 2nd Number BEFORE Swap: 20 6
1st Number and 2nd Number AFTER Swap: 6 20
```

VIVEKANANDA INSTITUTE OF PROFESSIONAL STUDIES - TECHNICAL CAMPUS Grade A++ Accredited Institution by NAAC
NBA Accredited for MCA Programme; Recognized under Section 2(f) by UGC; Affiliated to GGSIP University, Delhi; Recognized by Bar Council of India and AICTE An ISO 9001:2015 Certified Institution

SCHOOL OF ENGINEERING & TECHNOLOGY

EXPERIMENT:

Operators: Bitwise and sizeof

PROBLEM STATEMENT:

Use two unsigned int (or unsigned char) variables (say c & d), initialize them to some value of your choice. Do the following bitwise operations on those and print results as unsigned value on the screen:

c | d

c & d

c ^ d

~c

Verify the results printed with results of doing same bit wise operations on your own for your input values (of c and d).Additionally, print the sizeof variable c.

VERIFICATION:

c = 12 , d = 24

Binary of c and d

$c = (12)_{10} = (1100)_2$

$d = (24)_{10} = (11000)_2$

c & d

$01100 \;\&\; 11000 = (01000)_2 = (8)_{10}$

c | d

$01100 \mid 11000 = (11100)_2 = (28)_{10}$

~c

~01100 #unsigned int

$(11111111111111111111111111110011)_2$

$\Rightarrow (4294967283)_{10}$

c ^ d

$01100 \wedge 11000 = (10100)_2 = (20)_{10}$

C PROGRAM:

```c
// Operators: Bitwise and sizeof

#include <stdio.h>
int main()
{
    //Declaration
    unsigned int c, d;

    printf("Enter two numbers (say c and d) with space in between: ");
    scanf("%d %d", &c, &d);

    // sizeof c and d
    printf("c: %u\n",c);
    printf("Size of c: %u\n", sizeof(c));

    printf("d: %u\n", d);
    printf("Size of d: %u\n", sizeof(d));

    //Bitwise OR
    printf("Bitwise OR c|d %u\n", c|d);

    // Bitwise AND
    printf("Bitwise AND c&d %u\n", c&d);

    // Bitwise NOT
    printf("Bitwise NOT ~c %u\n", ~c);

    // Bitwise XOR
    printf("Bitwise XOR c^d %u\n", c^d);

    return 0;
}
```

OUTPUT:

```
c: 12
Size of c: 4
d: 24
Size of d: 4
Bitwise OR c|d 28
Bitwise AND c&d 8
Bitwise NOT ~c 4294967283
Bitwise XOR c^d 20
```

LEARNING OUTCOMES:

Explains the use of unsigned int for handling non-negative integer values, which is useful in bitwise operations.

Showcases how to get the memory size of variables using the sizeof operator, linking data types to memory usage.

Demonstrates bitwise logic: OR (|), AND (&), NOT (~), and XOR (^)

Combines user input and output for both binary operation results and memory size, making it easy to see the practical impact of C operators.