Certificate

Name: Shaurya Bhardwaj Class: Batch 20

Roll No: 59500023597 Exam No:

Institution UPES Bidholi

This is certified to be the bonafide work of the student in the _____ *Laboratory during the academic year 20 / 20 .*

No. of practicals certified _____ *out of* _____ *in the subject of* _____

.....
Teacher In-charge

.....
Examiner's Signature

.....
Principal

Date: .....

Institution Rubber Stamp

(N.B: The candidate is expected to retain his/her journal till he/she passes in the subject.)

Index

S. No.	Name of the Experiment	Page No.	Date of Experiment	Date of Submission	Remarks
1.	Experiment 1: Installation, Environment Setup and starting with C language:	1	6/08/25		
	1) Write a C program to print "Hello World"	01			
	2) Write a C program to print the address in multiple lines (new line)	02			
	3) Write a program that prompts the user to enter their name and age	03			
	4) Write a C program to add numbers, take number from user.	04			
2.	Experiment 2: Operators		6/08/25		
	1) WAP a C program to calculate the area and perimeter of a rectangle based on its length and width.	05			
	2) WAP a C program to convert temperature from Celsius to Fahrenheit using the formula: $F = (C * 9/5) + 32$	06			

Index

S. No.	Name of the Experiment	Page No.	Date of Experiment	Date of Submission	Rema
	Exp 3.1: Conditional Statements				
1)	To check triangle is valid or not and check if it is equilateral, right angle, scalene, isosceles.	7-8	13/08/25	20/8/25	
2)	To calculate BMI using its formula and health category	9-10	13/08/25	20/8/25	
	Exp 11: Bitwise operators				
1)	Program to apply bitwise OR, AND, NOT operator on bit level.	11-12	13/8/25	20/8/25	
2)	Program to apply left shift and right shift operator	13-14	13/8/25	20/8/25	
	Exp 3.2- ~~Variables~~ Loops	–		29/10/25	
1)	WAP to enter no. user wants. Display count of pos., neg, zero entered				
2)	WAP to print multiplication no.				
3)	WAP to generate given's output				
	Exp 4: Variable and Scope of Variable	–		29/10/25	
1)	Global variable inside and outside				
2)	Local variable inside a function				
3)	Declaring variables with diff^{n} codes blocks				
4)	Declare Static local variable				
	Exp 5: Array	–		29/10/25	
1)	Print largest integer number				
2)	Count and display pos, neg, odd even in array				
3)	Find freq^{n} of particular no.				
4)	Form Matrix A and B				

Index → PTO

Expt. No. _____ Page No. _____ Date _____

No	Name of Experiment	Pg No	Date of Experiment	Date of Submission	Remarks
8	Pointers Exp 8 1) Declare diff" type of pointers 2) Perform pointer arithmetic (increment and decrement 3) Function accepts pointers as variables	-	29/10/25		
6	Functions Exp 6 1) Develop a recursive and non-recursive 2) Develop a recursive func GCD	-	29/10/25		
7	Exp 7 → Structure & Union				
8	Experiment 9 → file Handling				
9	Experiment 10 → Dynamic memory allocation				
10	Experiment 12 → Preprocessor & Directives				
11	Experiment 13 → Macros in C				
12	Experiment 14 → Static library in C				

- Hello World

→ Code Execution Successful ←

Date

Expt. No. 01.1 Page No. 1

Experiment 1: Installation, Environment Setup and Starting with C language

1) Write a C program to print "Hello World"

- Coding:

```
1  #include <stdio.h>
3
4  int main() {
5      // Write C code here
6      printf("Hello World");
7      return 0;
8  }
```

- Output:

→ Hello World

- Address in multiple lines

main.cpp

```
#include <stdio.h>

int main() {

    printf("UPES BIDHOLI \n");
    printf("Dehradun Uttarakhand \n");
    printf("School of Computer Science \n");

    return 0;
}
```

Run

Output

```
UPES BIDHOLI
Dehradun Uttarakhand
School of Computer Science

=== Code Execution Successful ===
```

→ Code × Execution × Successful ×←

Date

Expt. No. _____ Page No. 2

2. Write a C program to print the address in multiple lines (new line)

★ Coding:

```
#include <stdio.h>
int main() {
// Write a C code to print address lines
printf ("UPES BIDHOLI \n");
printf ("Dehradun Uttarakhand \n");
printf ("School of Computer Science \n");

return 0;
}
```

★ Output:

- UPES BIDHOLI
- Dehradun Uttarakhand
- School of Computer Science

Date :

Expt. No. Page No. 4

4) Write a C program to add numbers, take numbers from User....

☆ Coding:

```
#include main () {
// Addition of Two numbers

int a, b, sum;
printf ("The first number is \n");
scanf ("%d", &a);
printf ("The second number is \n");
scanf ("%d", &b);
sum = a+b
printf ("The sum of the number is %d", sum);

return 0;
}
```

☆ Output:

- The first number is
  40
- The second number is
  60
- The sum of the numbers is 100

Teacher's Signature

• Area and Perimeter of a Rectangle

= Code Execution Successful =

Experiment 2: Operators

1) WAP a C program to calculate the area and perimeter of a rectangle based on its length and width.

☆ Coding:

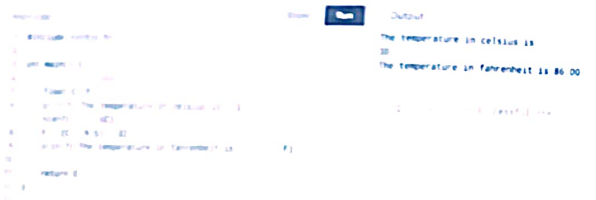
```
1  #include<main> {}
2  // Calculating area and perimeter of a rectangle
3  int main() {
4  float a, b, area, perimeter;
5  printf("The length of rectangle is \n");
6  scanf("%f", &a);
7  printf("The breadth of rectangles is \n");
8  scanf("%f", &b);
9  area = a*b;
10 perimeter = 2*(a+b);
11 printf("The area of rectangle is %.2f \n", area);
12 printf("The perimeter of rectangle is %.2f \n", perimeter);
13 return 0;
14 }
```

☆ Output

- The length of Rectangle is 10
- The Breadth of Rectangle is 12
- The area of rectangle is 120.00
- The perimeter of rectangle is 44.00

Teacher's Signature

• Converting Temp from Celsius to Fahrenheit



→ Code Execution Successful ←

Date

Expt No. _____ Page No. 06

2. WAP a C program to convert temperature from Celsius to Fahrenheit using formula F = (C*9/5) + 32

★ Coding:

```
1. #include <stdio.h>
2. int main() {
3. // Converting Celsius to fahrenheit using C program
4. float C,F;
5. printf("The temperature in celsius is \n");
6. scanf("%f",&C);
7. F= (C*9/5)+32;
8. printf("The temperature in fahrenheit is %.2f \n",F);
9.
10. return 0;
11. }
```

Output

The temperature in celsius is
30
The temperature in fahrenheit is 86.00

Teacher's Signature _____

Date

Expt. No. Page No. 04

Experiments 3.1 : Conditional Statements:

1) WAP to take check if the triangle is valid or not. If the validity is established, do check if the triangle is isoceles, right angle, or scalene. Take sides of the triangle as input from a user.

Steps

- Step 1: Take three sides of the triangle as input from the user
- Step 2: Check if the Triangle Inequality Theorem to verify if the triangle is valid
- Step 3: If valid, check if it is Equilateral, Isoceles, or Scalene using equality conditions
- Step 4: Check pythagoras theorem to determine if it is a Right-Angled triangle

Teacher's Signature

Algorithm

- Input three sides of the triangle from the user
- Check validity using Triangle Inequality Theorem
  i.e. (a+b>c), (a+c>b), (b+c>a)
- If valid, determine the type:
  All equal → Equilateral
  Two equal → Isosceles
  All different → Scalene
- Additionally, check if it satisfies the Pythagoras theorem for right-angled triangle

Here, Output:

→ Enter three sides of triangle: 3 4 5

→ Triangle is valid

→ It is a scalene triangle

→ It is also a right-angled triangle

Date

Expt No. Page No. 08

Input:

```
#include <stdio.h>
int main () {
  float a, b, c;
  // Input sides
  printf ("Enter three sides of triangle: ");
  scanf ("%f %f %f", &a, &b, &c);
  // Check triangle validity
  if ((a+b>c) && (a+c>b) && (b+c>a)) {
  printf ("Triangle is valid.\n");
  // Check triangle type
  if (a==b && b==c) {
  printf ("It is an equilateral triangle.\n"); }
  else if (a==b || b==c || a==c){
  printf ("It is an isosceles triangle.\n"); }
   else {
  printf ("It is a Scalene triangle.\n"); }
  // Check right angle
  if ((a*a == b*b + c*c) || (b*b == a*a + c*c) ||
       (c*c == a*a + b*b)) {
  printf ("It is also a Right-Angled triangle.\n"); }
   } else {
  printf ("Triangle is not valid.\n"); }
  return 0;
  }
```

Teacher's Signature

Date

Expt. No. Page No. 09

2. WAP to compute the BMI index of a person and print BMI value as per the following ranges.
BMI = Weight (kg)/Height^2 (mts)

Steps

1) Start the Program by declaring variables for weight, height and BMI. Ask the user to enter weight in kgs and height in meters. Store value for calculation

2) Apply the BMI formula:

$$BMI = \frac{\text{Weight (kg)}}{\text{Height (m)} \times \text{Height (m)}}$$

- < 15 → Starvation
- 15.1 to 17.5 → Anorexic
- 17.6 to 18.5 → Underweight
- 18.6 to 24.9 → Ideal
- 25 to 25.9 → Overweight
- 30 to 39.9 → Obese
- 40 + → Morbidly Obese

3) Display Output: Showing both the numeric BMI value (rounded to 2 decimal values) and health category. End the program.

Teacher's Signature

Algorithm

- Input weight (in kilograms) and height (in meters) from user
- Calculate BMI using formula
  BMI = weight (kg) / height (m) * Height (m)
- Display the corresponding health category

Output:

=> Enter weight (in kg): 70
=> Enter height (in meters): 1.75
=> Your bmi is: 22.86
=> Category: Ideal

Date

Expt. No.

Page No. 010

Input:

```
#include <stdio.h>
int main () {
Float weight, height, bmi;
  // Input weight and height
printf ("Enter weight (in kg): ");
Scanf ("%f", &weight);
printf ("Enter height (in meters): ");
scanf ("%f", &height);
  // Calculate BMI
bmi = weight / (height * height);
printf ("Your BMI is: %.2f \n", bmi);
// Determine BMI category
if (bmi < 15)
printf ("Category: Starvation \n");
else if (" Category bmi >= 15.1 && bmi < 17.5)
 printf ("Category: Under weight \n");
else if (bmi >= 17.6 && bmi <= 24.9)
printf ("Category: Ideal \n");
else if (bmi >= 25 && bmi <= 29.9)
printf ("Category: Obese \n");
else if (bmi > 30 && bmi < 39.9)
printf ("Category: Morbidly obese \n");

  return 0;
}
```

Date

Expt. No. 11 Page No. 11

Experiment No. 11: Bitwise operators

1) Write a program to apply bitwise OR, AND, NOT operators on bit level.

Steps

1) Start the program and take two integer inputs from the user
2) Apply Bitwise AND (&) and OR (|) operators between the two numbers
3) Apply bitwise NOT (~) operator on both numbers seperately.
4) Display all the results on the screen.

<Algorithm>

Input two numbers: a and b

Compute a & b, a|b and ~a, ~b

Store results in variables

Print the results and end the program

Expt. No. _____ Page No. _____

Input:

```
#include <stdio.h>
int main () {
        // bit level operator
int a = 5;
int b = 7;

printf ("Bitwise a or b is %d\n", a|b);
printf ("Bitwise a and b is %d\n", a&b);
printf ("Bitwise not of a is %d\n", ~a);
return 0;
}
```

Output:

```
bitwise a or b is 7
bitwise a and b is 5
bitwise not of a is -6

=> Code Execution Successful <=
```

Date

Expt. No. 11.2

Page No. 13

2) Write a program to apply left shift and right shift operators.

→ Steps ←

1. Start the program and take an integer input from the user
2. Ask the user how many positions to shift.
3. Apply left shift (<<) and right shift (>>) on the number.
4. Display the results.

→ Algorithm ←

Input number n and shift value s

↓

Compute n << s (left shift result)

↓

Compute n >> s (right shift result)

↓

Print both results and end the program

Date

Expt. No. Page No. 14

Input:

```
#include <stdio.h>

int main () {
int a = 5;
int b = 7;

printf("The left shift operator of a %d \n", a<<1);
printf("The left shift operator of a %d \n", a<<1);
printf("The right shift operator of a %d \n", a>>1);
printf("The right shift operator of a %d \n", a>>1);
return 0; }
```

Output:

```
→ The left shift operator of a 10
→ The left shift operator of a 10
→ The right shift operator of a 2
→ The right shift operator of a 2
```

= = Code Execution Successful = =

Steps:

Keep taking input numbers

Count positive, negative, zero entries

Display the total counts at the end

Expt No. _____ Page No. _____

Experiment no: 3.2 : Loops

1) WAP to enter numbers till the user wants. At the end, it should display the count of pos^t, neg^t, zero entered

```
#include <stdio.h>

int main() {
int num;
int positive = 0, negative = 0, zero = 0,
char choice;

printf ("---- Count positive, Negative, and zero
        Numbers ---- \n");

do {
   printf ("Enter a Number:" );
   scanf ("%d", &num);

   if (num > 0)
   positive ++;
  else if (num < 0)
   negative ++;
  else
   zero++;

Printf (" Do you want to enter another number ? (Y N) :");
Scanf (" %c", &choice); }
```

Date

Expt. No.

Page No.

2) WAP to print the multiplication table of the number entered by the user. It should be in the correct formatting. Num * 1 = Num

```
#include <stdio.h>
int main() {
int num, i;

printf("---- Multiplication Table ---- \n");
printf("Enter a number: ");
scanf("%d", &num);

printf("\n Multiplication Table of %d: \n", num);
for (i=1; i<=10; i++) {
  printf("%d * %d = %d \n", num, i, num*i); }
  return 0;
    }
```

Output:

```
Enter a number: 7
  7 * 1 = 7
  7 * 2 = 14
  7 * 3 = 21
  ...
  7 * 10 = 70
```

Date

Expt. No. Page No.

2) WAP to print the multiplication table of the number entered by the user. It should be in the correct formatting. Num * 1 = Num

```
#include <stdio.h>
int main() {
int num, i;

printf("---- Multiplication Table ---- \n");
printf("Enter a number: ");
scanf("%d", &num);

printf("\n Multiplication Table of %d: \n", num);
for (i=1; i<=10; i++) {
  printf("%d * %d = %d \n", num, i, num*i); }
  return 0;
  }
```

Output:

```
Enter a number: 7
 7*1 = 7
 7*2 = 14
 7*3 = 21
 ....
 7*10 = 70
```

Steps:

Print pattern like Pascal's triangle using loops

Use nested loops for rows and columns.

Date

Expt. No. Page No.

3) WAP to generate the following set of output:

a)
```
1
2 3
4 5 6
```
⇒ Output

```
#include <stdio.h>
int main() {
int i, j, num = 1;

printf("---- Pattern (a) ----\n");

for (i = 1; i <= 3; i++) {
for (j = 1; j <= i; j++) {
printf("%d", num);
num++; }
printf("\n"); }
return 0;
}
```

Date _____

Expt. No. _____ Page No. _____

b)

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

=> Output

```
# include <stdio.h>
int main() {
int n= 5, i, j, num;
printf (" ---- Pascal's Triangle Pattern ---- \n");
for (i=0; i<n; i++) {
num = 1;
for (j=0; j<=i; j++) {
print ("%d", num);
num= num* (i-j)/(j+1);
}
print ("\n"); }
return 0;
}
```

Steps:

Start with 100000

Use a loop to increased by 10% each year for 10 years

Display population at the end of each year

Output:

Year 1: 110000
Year 2: 121000
Year 3: 133100
Year 4: 146410
Year 5: 161051
Year 6: 177156
Year 7: 194871
Year 8: 214358
Year 9: 235794
Year 10: 259374

Date

Expt. No. Page No.

4) The population of a town is 100000. The pop^n has increased steadily at the rate of 10% per year for last 10 years write a program to determine the population at end of each year in last decade.

```
#include <stdio.h>
int main() {
float population = 100000.;
int year;

printf(" ---- Population Growth ---- \n");
for (year = 1; year <= 10; year++) {

population = population + (population * 0.10);
printf(" Population at end of year %d = %f \n",
year, population);
}
return 0;
}
```

Steps:

Declare a variable outside all functions.

Access and modify it inside multiple functions

Observe that changes are shared among all functions.

Output:

Inside main: x = 10

Inside funx^n: x = 20

Back in main: x = 20

Date

Expt. No. Page No.

Experiment 4: Variable and Scope of Variable:

i) Declare a global variable outside all functions and use it inside various functions to understand its accessibility.

```
#include <stdio.h>

int num = 10;   // Global variable

void show () {
printf ("Inside function show(), num = %d\n",
num);
}

int main () {
printf ("Inside main() num = %d \n", num);
show();
num = num + 5;
printf ("After changing, num = %d \n", num);
show();
return 0; }
```

Output: Inside main (), num = 10
Inside function show (), num = 10
After changing, num = 15
Inside function show(), num = 15

Teacher's Signature

Steps:

Declare a variable inside fun^n

Try to use it outside

Compare with global variable scope.

Output:

Inside function: a = 5

~~Error: 'a' undeclared (out of function)~~

Date

Expt. No. _____ Page No. _____

2.) Declare a local variable inside a function and try to access it outside the function. Compare this with accessing the global variable from within the function.

```
#include <stdio.h>

void test() {
int x = 5;
printf("Inside test(), x = %d\n", x);
}
int main() {
test();
printf("Local variable x can't be accessed here\n");
return 0; }
```

Output:

Inside test(), x = 5

Local variable x can't be accessed here.

Steps:

Declare variable outside and inside block

Print values inside and outside block

Observe which variables are accessible

Output:

Inside block: y: 30

Outside block: y is not accessible

Date

Expt No. Page No.

3) Declare variables within different code blocks (enclosed by curly braces) and test their accessibility within and outside those blocks.

```
#include <stdio.h>

int main () {
int a = 10;
printf (" Outside block: a = %d\n", a);
  {
  int b = 20;
printf (" Inside block: a= %d, b= %d\n",a,b);
  }
printf (" Variable b can't be accessed outside
          the block.\n");
return 0;
  }
```

Outside:

Outside block: a=10

Inside block: a=10, b=20

Variable b can't be accessed outside the block.

Steps:

Declare static variable inside fun'n

Call function multiple times.

Observe that value persists btw calls.

Output:

Function called, count = 1

Function called, count = 2

Function called, count = 3

Expt No. _____ Date _____ Page No. _____

4) Declare a static local variable inside a function. Observe how its value persists across function calls.

```
#include <stdio.h>

Void counter() {
Static int count = 0;
Count ++;
printf ("Count = %d \n", count); }

int main () {
counter ();
counter ();
counter ();
printf (" Static variable keeps it value between
        call. \n")
return 0;
}
```

Output:

Count = 1

Count = 2

Count = 3

Static variable keeps its value between calls.

Steps:

Input an array of integers

Find largest and second largest using a loop.

Display the second largest number

Output:

~~Enter number of elem~~

Enter how many numbers: 10 20 30 40 50

Second largest = 40

Date

Expt No. Page No.

Experiment 5: Array

1) Write to read a list of integers and store it in a single dimensional array. Write to a C program to print second ~~least~~ largest integer in list of integers.

```
#include <stdio.h>

int main() {
int n, i, largest, second,
int arr [100];

printf ("--- Find second largest number ---- \n");
printf (" Enter how many number: ")
scanf ("%d", &n);

printf (" Enter %d numbers: \n", n);
for (i = 0; i<n; i++) {
scanf (" %d ", &arr[i]); }

largest = second = arr [0];
for (i = 1; i < n ; i++ ) {
if (arr [i] > largest ) {
    second = largest;
    largest = arr [i]; }
```

Date _____

Expt. No. _____ Page No. _____

```
else if(arr[i] > second && arr[i] < largest){
    second = arr[i];
    }
}
printf("Second largest number = %d\n", second);
return 0;
}
```

Steps:

Read all elements in array

Use loops and condition to count

Print all counts

Output:

Enter no of elements. 3 -5 6 0 8

Positive numbers: 3

Negative numbers: 1

Even number = 3

Odd number = 2

Expt No. _____ Date _____

Page No. _____

2. WAP to read a list of integers and store in single dimensional array. Write a C program to count and display positive, negative, odd, even numbers in an array.

```
#include <stdio.h>
int main() {
int n, i, num;
int positive = 0, negative = 0, even = 0, odd = 0;
int arr [100];

printf (" Enter no. of elements: ");
scanf ("%d", &n);

printf (" Enter %d numbers: \n", n);
for (i = 0; i < n; i++) { scanf ("%d", &arr [i]); }

for (i = 0; i < n; i++ ) {
if (arr [i] > 0)
    positive ++;
else if (arr [i] < 0)
    negative ++;
if (arr [i] % 2 == 0)
    even ++
else
    odd ++
}
```

Teacher's Signature _____

Date

Expt. No. Page No.

```
printf ("\n Positive numbers = %d\n", positive);
printf ("Negative numbers = %d\n", negative);
printf ("Even numbers = %d\n", even);
printf ("odd numbers = %d\n", odd);

return 0;
}
```

Steps:

Read array elements and numbers to check

Count how many times that number appears

Print Frequency

Output: n

Enter no of Elements : 2 4 6 2 4 2

Enter number to find frequency: 2

Frequency of 2 = 3

Date

Expt. No. Page No.

3) Write a C program to find frequency of a particular no. in a list of integers.

```
#include <stdio.h>

int main() {
int n, i, search, count = 0;
int arr[100];

printf("Enter no. of Elements: ");
scanf("%d", &n);

printf("Enter %d numbers: \n", n);
for (i = 0; i < n; i++) {
scanf("%d", &arr[i]); }

printf("Enter the number to find frequency: ");
scanf("%d", &search);

for (i = 0; i < n; i++) {
  if (arr[i] == search)
  count + +; }

printf("Frequency of %d = %d times\n", search,
        count);
return 0;
}
```

Teacher's Signature

Steps:

Input two matrices A and B

Check if multiplication is possible

Multiply and display result in matrix form

Output:

Enter rows and columns of A: 2 2
Enter rows and columns of B: 2 2
Enter elements of matrix A:
1 2
3 4
Enter elements of matrix B:
5 6
7 8
Resultant matrix
19 22
43 50

Date

Expt. No. _____ Page No. _____

1) WAP that reads two matrices A (m x n) and B (p x q) and computes the product A and B. Read matrix A and matrix B in row major order respectively. Print both the input matrices and resultant matrix in format only. Program must check the compatibility of orders of matrices for multiplication. Report appropriate message in case of incompatibility.

```
#include <stdio.h>
int main () {
int a[10][10], b[10][10], result[10][10];
int m, n, p, q, i, j, k;

printf ("Enter rows and columns of matrix A: ")
scanf ("%d %d", &m, &n);
printf ("Enter rows and columns of matrix B: "),
scanf ("%d %d", &p, &q);

if (n != p) {
printf ("Matrix multiplication not possible \n");
return 0;
}
printf (" Enter elements of matrix A: \n");
for (i=0; i < m ; i++) {
   for (j=0; j < n ; j++) {
   scanf ("%d", & a[i][j]); }
}
```

Teacher's Signature _____

Expt. No. _____ Date _____

Page No. _____

```
printf("Enter elements of matrix B:\n");
for(i=0; i<p; i++) {
for(j=0; j<q; j++) {
scanf("%d", &b[i][j]); }
}
for(i=0; i<m; i++) {
 for(j=0; j<q; j++) {
  result[i][j]=0; }
}
for(i=0; i<m; i++) {
for(j=0; j<q; j++) {
for(k=0; k<n; k++) {
result[i][j] += a[i][k]* b[k][j];
}
  }
   }
printf("\n matrix A:\n");
for(i=0; i<m; i++) {
for(j=0; j<n; j++) {
printf("%d", a[i][j]); }
 printf("\n");
  }
printf("\n Resultant Matrix
 printf("\n matrix B:\n");
for(i=0; i<p; i++) {
 for(j=0; j<q; j++) {
  printf("%d", b[i][j]); }
  printf("\n");
   }
```

Teacher's Signature _____

Expt. No. _____ Date _____

Page No. _____

```
printf("\n Resultant Matrix (A x B):\n");
for(i=0; i<m; i++){
for(j=0; j<q; j++){
printf("%d", result[i][j]);
}
printf("\n");
}
return 0;
}
```

Steps:

Declare int, float, char variables

Create pointers for each type and store their addresses

Print pointer values and the data they point to.

Output:

Value of a = 10, Address = 0x7ffd8c1c

Value of b = 20.5, Address = 0x7ffd8c20

Value of c = X, Address = 0x7ffd8c24

Date

Expt. No.

Page No.

Experiment 8 : Pointers

1) Declare different types of pointers (int, float, char) and initialize them with the addresses of variables. Print the values of both pointers and variables they point to.

```
#include <stdio.h>
int main () {

int a=10;
float b = 20.5;
char c = 'X';

// Declare the pointers
int *ptr1 = &a;
float *ptr2 = &b;
char *ptr3 = &c;

printf ("Value of a : %d, Address of a = %p,
        Pointers value = %p\n", a, &a, ptr1);
printf ("Value of b = %.2f, Address of b = %p,
        Pointers value = %p\n", b, &b, ptr2);
printf ("Value of c = %c, Address of c = %p, Pointer
        value = %p\n", c, &c, ptr3);

// Printing value using pointers
Printf ("\n Accessing value through pointers: \n");
```

Teacher's Signature

Expt. No. _____ Date _____

Page No. _____

```
printf("*ptr1 = %d\n", *ptr1);
printf("*ptr2 = %.2f\n", *ptr2);
printf("*ptr3 = %c\n", *ptr3);

return 0;
}
```

Steps:

Declare pointers of int, float, char

Increment and decrement them.

Observe how address changes for each data type.

=

Output:

Original

Date

Expt No.

Page No.

2) Perform pointer Arithmetic (increment and decrement) on pointers of different data types. Observe how the memory addresses change and effects on data access.

```
#include <stdio.h>

int main() {

int a = 5;
float b = 10.5;
char c = 'A';

int *ptr1 = &a;
float *ptr2 = &b;
char *ptr3 = &c;

printf("Original address of ptr1(int): %p\n",
        ptr1);
 ptr1++;
printf("After ptr1++ = %p \n", ptr1);
ptr1--;
printf("After ptr1-- (back) = %p\n", ptr1);

printf("\n Original address of ptr2 (float) =
        %p\n", ptr2);
```

Date

Page No.

Expt. No.

```
ptr2++;
printf("After ptr2++ = %p\n", ptr2);
ptr2--;
printf("After ptr2-- (back) = %p\n", ptr2);
printf("\n Original address of ptr3 (char)
        = %p\n", ptr3);
ptr3++;
printf = ("After ptr3++ = %p\n", Ptr);
ptr3--;
printf("After ptr3-- (back) = %p\n", Ptr3);

return 0;
}
```

Steps:

Write a funcn that takes pointers as parameters

Pass variables by reference using pointers.

Modify their values inside the funcn and print updated results.

Output:

Before Modifying: x = 10, y = 15.5

After modifying: x = 15, y = 10

Expt. No. _____ Date _____ Page No. _____

3) Write a function that accepts pointers as parameters. Pass variables by reference using pointers and modify their values within the function.

```
#include <stdio.h>

void modifyValues(int *x, float *y, char *z){
*x = *x + 5;
*y = *y + 2.5;
*z = *z + 1;
}

int main(){
int a = 10;
float b = 15.5;
char c = 'B';

printf("Before Modifying \n");
printf("a = %d, b = %.2f, c = %c \n", a, b, c);

modifyValues(&a, &b, &c);

printf("\n After modifying \n");
printf("a = %d, b = %.2f, c = %c \n", a, b, c);

return 0;
}
```

Teacher's Signature _____

Steps:

Take input n and r

Use recursive and non-recursive function to find factorial

Use formula nCr = n! / (r! * (n-r)!) to find binomial coefficient

Output:

Enter n and r: 5 2

Binomial coeff^n (nCr) = 10.00

Recursive factorial of 5 = 120

Non-Recursive factorial of 5 = 120

Expt. No. _____ Date _____ Page No. _____

Experiment - 6 . Functions:

1) Develope a Recursive and non-recursive function to compute the binomial coeff^n

```
#include <stdio.h>
int fact_rec (int n) {
if (n == 0)
return 1;
else
return n * fact_rec (n-1); }

int fact_nonrec (int n) {
int i, fact = 1;
for (i=1; i<=n, i++) {
    fact = fact * i;          }
    return fact; }

int main () {
int n, r;
float binomial;

printf ("Enter n and r:");
scanf ("%d %d", &n, &r);

binomial = (float) fact_rec(n) / (fact_rec (r) *
                    fact_rec (n-r));
printf ("Binomial coefficient (nCr) : %.2f \n",
                    binomial);
```

Teacher's Signature _____

Expt. No. _____ Date _____

Page No. _____

```
printf("Recursive factorial of %d = %d\n", n,
        fact_rec(n));
printf("Non-Recursive Factorial of %d =
        %d\n", n, fact_nonrec(n));
return 0;
}
```

Steps

Take two numbers as input

Use recursion with condition gcd(a,b) = gcd(b, a%b)

Display the GCD of two numbers

Output:

Enter two numbers: 24 36
GCD of 24 and 36 is 12

Expt No. _____ Date _____ Page No. _____

2) Develop a recursive function GCD(num1, num2) that accepts two integer arguments. Write a C program that invokes this "func" to find greatest common divisor of two given integers.

```
#include <stdio.h>
int gcd(int a, int b) {
  if (b == 0)
    return a;
  else
    return gcd(b, a%b);
}
int main() {
  int num1, num2;
  printf("Enter two numbers:");
  scanf("%d %d", &num1, &num2);
  printf("GCD of %d and %d is %d\n", num1, num2,
         gcd(num1, num2));
  return 0;
}
```

Date

Experiment 7: Structures & Union:

Complex number:

7.1

```
#include <stdio.h>
struct complex {
 float real, imag; };
struct Complex readcomplex () {
   struct Complex c;
 printf (" Enter real part :");
scanf (" %f", &c.real);
printf (" Enter imaginary part: ");
scanf (" %f", &c.imag); return c; }
void printComplex (struct Complex c) {
printf (" %.1f + %.1f i\n", c.real, c.imag); }
int main () {
struct Complex c1, c2, sum, diff;
c1 = readComplex (); c2 = readComplex ();
sum.real = c1.real + c2.real;
sum.imag = c1.imag + c2.imag;
diff.real = c1.real - c2.real;
diff.imag = c1.imag + c2.imag;
printf (" Sum = ");
printComplex (sum);
printf (" Difference =");
printComplex (diff);

return 0;
}
```

Date

Employ Salary:

7.2

```
#include <stdio.h>
struct employee {
char name[50];
float basic, da, gross;  };
int main () {
struct Employee emp[100];
int i;

for (i=0; i<100; i++) {
printf ("Enter name:");
scanf ("%s", emp[i].name);

printf ("Enter basic pay:");
scanf ("%f", &emp[i].basic);

emp[i].da = 0.52* emp[i].basic;
emp[i].gross = emp[i].basic + emp[i].da;

printf ("Gross Salary of %s = %.2f\n", emp[i].nam
                                      ,emp[i].gross);
}
return 0;
}
```

Date

7.3 Book Details

```
#include <stdio.h>
struct Book {
int id;
char title [50];
char author [50];
float price;
};
void printBook (struct Book b) {
printf ("ID: %d\n", b.id);
printf ("Title: %s\n", b.title);
printf ("Author: %s\n", b.author);
printf ("Price: %.2f\n", b.price);
}
int main () {
struct Book b1;
printf ("Enter book id: ");
Scanf ("%d", &b1.id);
printf ("Enter title: ");
Scanf ("%s", b1.title);
printf ("Enter author: ");
Scanf ("%s", b1.author);
printf ("Enter price: ");
Scanf ("%f", &b1.price);
printBook (b1);

return 0;
}
```

9.2

Date

7.4 Union- Display Address

```
#include <stdio.h>
union address {
char name [50];
char home [50];
char hostel [50];
char city [50];
char state [50];
char zip [10];
};
int main () {
union Address a;

printf(" Enter your present address (city): ");
Scanf ("%s", a.city);

print f (" Your present address is: %s\n", a.city);

return 0;
}
```

Date

Experiment 9 - FILE Handling

9.1 Create file & write text

```
#include <stdio.h>
int main() {
file *fp;
fp = fopen ("myfile.txt", "w");
if(fp == NULL) {
  printf ("Error opening file!");
return 0; }
fprint (fp, "This is a beginner C file.\n");
fclose (fp);
printf ("file written successfully:");
return 0; }
```

9.2 Read file character by character

```
#include <stdio.h>
int main () {
file *fp; char ch;
fp = fopen ("myfile.txt", "r");
if (fp == Null) { printf ("file not found!"); }
while ((ch = fgetc (fp)) != EOF) {
  printf ("%c", ch); }
fclose (fp);
return 0;
}
```

Date

9.3 Read file line by line

```
#include <stdio.h>
int main() {
FILE *fp;
char line [100];
fp = fopen ("myfile.txt", "r");
if (fp == NULL) {
printf ("file not found!");
return 0; }
while(fgets(line, sizeof(line), fp)) {
printf ("%s", line); }
fclose (fp);
return 0; }
```

Date

Experiment 10 Dynamic Memory Allocation

10.1 Create Simple Linked List

```
#include <stdio.h>
#include <stdlib.h>
Struct Node {
int data;
Struct Node *next; };
int main () {
Struct Node *head = NULL, *temp, *NewNode;
int i, n;
printf ("Enter no of Nodes:"); scanf ("%d", &n);
for (i = 0; i<n; i++) {
newNode = (Struct Node*) malloc (sizeof (Struct Node));
printf (" Enter data:");
Scanf ("%d", &newNode -> data);
new Node -> next = NULL;
if (head == NULL) { head = newNode; } else{
temp=head; while (temp -> next != NULL)
temp = temp -> next; temp -> next = newNode; }}
temp = head;
printf (" Linked List:");
while (temp != NULL) {
printf ("%d", temp -> data);
temp = temp -> next; }
return 0;
}
```

Date

10.2 Insert in the middle:

```
#include <stdio.h>
#include <stdlib.h>
struct Node { int data;
struct Node* next; };
int main () {
struct Node* head = NULL, *temp, *NewNode; int pos, value, i;
for (i = 1; i <= 3; i++) {
newNode = (struct Node*) malloc (sizeof (struct Node));
newNode -> data = i; newNode -> next = head;
head = newNode;
printf (" Enter position to insert:"); scanf ("%d", &pos);
printf (" Enter Value :");
scanf ("%d", &value);
newNode = (struct Node*) malloc (sizeof (struct Node));
newNode -> data = value;
temp = head; for(i = 1; i < pos - 1; i++)
  temp = temp -> next;
newNode -> next = temp -> next; temp -> next = newNode;
temp = head;
printf ("Updated list :");
while (temp != NULL) {
printf ("%d", temp -> data);
temp = temp -> next;
}
return 0;
}
```

Date

Experiment 12: Preprocessor and Directives in C

12.1 Define Constant

```
#include <stdio.h>
#define PI 3.14
int main() {
printf("Value of PI = %.2f", PI);
 return 0; }
```

12.2 Function-like Macro

```
#include <stdio.h>
#define SQUARE(x) (x*x)

int main() }
int a;
printf("Enter number:");
scanf("%d", &a);
printf("Square = %d", SQUARE(a));
  }
```

Date

Experiment 13: Macros in C

13.1: Arithmetic functions:

```
#include <stdio.h>
#define ADD(x,y));
#define SUB(x,y));
#define MUL,
#define DIV,
#include <stdio.h>
#define ADD(a,b) (a+b)
#define SUB(a,b) (a-b)
#define MUL(a,b) (a*b)
#define DIV(a,b) (a/b)

int main() {
int x=10, y=5;

printf("Add = %d\n", ADD(x,y));
printf("Sub = %d\n", SUB(x,y));
printf("Mul = %d\n", MUL(x,y));
printf("Div = %d\n", DIV(x,y));
return 0;
}
```

Date

Experiment 14 - Static Library

arith.c

```
int add (int a, int b) { return a + b; }
int sub (int a, int b) { return a - b; }
int mul (int a, int b) { return a * b; }
int div (int a, int b) { return a / b; }
```

main.c

```
#include <stdio.h>
int add (int, int);
int sub (int, int);
int mul (int, int);
int div (int, int);

int main() {
printf("%d\n", add (2,3));
return 0;
}
```