ADVANCED PROGRAMMING LANGUAGE – 'C' PRACTICAL

PRACTICAL 1: Write a program to check the given number is Palindrome or not using User Defined Function (UDF).

```c
#include <stdio.h>

void pal(int x)
{
    int rem, rev = 0, temp = x;
    while (x > 0)
    {
        rem = x % 10;
        rev = rev * 10 + rem;
        x = x / 10;
    }

    if (rev == temp)
    {
        printf("Given Number is palindrome");
    }
    else
    {
        printf("Given Number is not palindrome");
    }
}

void main()
{
    int n;
    printf("Enter a Number:");
    scanf("%d", &n);
    pal(n);
}
```

output:

```
Enter a Number:121
Given Number is palindrome
```

PRACTICAL 2: Write a program to find factorial of given no using UDF.

```
#include <stdio.h>

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

void main()
{
    int n, factorial;
    printf("Enter a Number:");
    scanf("%d", &n);
    factorial = fact(n);
    printf("Factorial of given number is: %d\n", factorial);
}
```

output:

```
Enter a Number:4
Factorial of given number is: 24
```

PRACTICAL 3: Write a program to find factorial of given no using recursion.

```c
#include<stdio.h>

    int fact(int n){
        if(n>=1)
            return n*fact(n-1);
        else
            return 1;
    }
    void main(){
        int n, factorial;
        printf("Enter a Number:");
        scanf("%d",&n);
        printf("Factorial of given number is: %d\n", fact(n));
    }
```

output:

```
Enter a Number:5
Factorial of given number is: 120
```

PRACTICAL 4: Write a program to display first 25 terms of Fibonacci series using recursion.

```c
#include <stdio.h>

int checkFib(int num)
{
    if (num == 0)
    {
        return 0;
    }
    else if (num == 1)
    {
        return 1;
    }
    else
    {
        return checkFib(num - 1) + checkFib(num - 2);
    }
}
void main()
{
    int num = 25, i;
    printf("Fibonacci series:");
    for (i = 0; i < num; i++)
    {
        printf("%d ", checkFib(i));
    }
}
```

output:

```
Fibonacci series:0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597
2584 4181 6765 10946 17711 28657 46368
```

PRACTICAL 5: Write a program using a recursive function to find the GCD (Greatest Common Divisor) of two Positive integer numbers.

```c
#include <stdio.h>

int gcd(int a, int b) {
    if (b == 0) {
        return a;
    } else {
        return gcd(b, a % b);
    }
}
void main() {
    int num1, num2;

    printf("Enter two positive integers: \n");
    scanf("%d \n %d", &num1, &num2);

    printf("GCD of %d and %d is %d", num1, num2, gcd(num1, num2));

}
```

output:

```
Enter two positive integers:
10
20
GCD of 10 and 20 is 10
```

PRACTICAL 6: Write a program to swap value of two integer number using UDF.

```
#include <stdio.h>

void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

void main() {
    int x, y;

    printf("Enter two integer numbers:\n");
    scanf("%d %d", &x, &y);

    printf("Before swapping: x = %d, y = %d\n", x, y);

    swap(&x, &y);

    printf("After swapping: x = %d, y = %d\n", x, y);

}
```

output:

```
Enter two integer numbers:
10
15
Before swapping: x = 10, y = 15
After swapping: x = 15, y = 10
```

PRACTICAL 7: Write a function prime that returns 1 if its argument is a prime and return zero Otherwise.

```
#include <stdio.h>

int prime(int n) {
    if (n < 2) {
        return 0;
    }
    for (int i = 2; i * i <= n; i++) {
        if (n % i == 0) {
            return 1;
        }
    }
    return 1;
}
```

output:

PRACTICAL 8: Write a program that uses a UDF to sort an array of integer.

```c
#include <stdio.h>

// User-Defined Function (UDF) to sort an array of integers
void arrSort(int arr[], int n)
{
    int i, j, temp;
    for (i = 0; i < n - 1; i++)
    {
        for (j = 0; j < n - i - 1; j++)
        {
            if (arr[j] > arr[j + 1])
            {
                // swap arr[j] and arr[j+1]
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
}

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

    // ask user for the size of the array
    printf("Enter the size of the array: ");
    scanf("%d", &n);

    // ask user to input the elements of the array
    printf("Enter %d integers:\n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%d", &arr[i]);
    }

    // call the UDF to sort the array
    arrSort(arr, n);

    // print the sorted array
    printf("Sorted array:\n");
    for (i = 0; i < n; i++)
    {
        printf("%d ", arr[i]);
    }
    printf("\n");
}
```

output:

```
Enter the size of the array: 5
Enter 5 integers:
2
1
5
3
4
Sorted array:
1 2 3 4 5
```

PRACTICAL 9: Write a program which explains the use of nesting of functions.

```c
#include <stdio.h>

void outerFunction(int x)
{
    void innerFunction(int y)
    {
        printf("The value of x is %d\n", x);
        printf("The value of y is %d\n", y);
    }

    int y = x * 2;
    innerFunction(y);
}

void main()
{
    int x = 5;
    outerFunction(x);
}
```

output:

```
The value of x is 5
The value of y is 10
```

PRACTICAL 10: Define a structure type struct personal that would contain person name, date of joining and salary using this structure to read this information and Display on screen.

```c
#include <stdio.h>


    struct personal
    {
        char name[50];
        char date_of_joining[20];
        float salary;
    };

    void main()
    {
        struct personal p;

        printf("Enter person name: ");
        scanf("%s", p.name);

        printf("Enter date of joining (DD/MM/YYYY): ");
        scanf("%s", p.date_of_joining);

        printf("Enter salary: ");
        scanf("%f", &p.salary);

        printf("Person name: %s\n", p.name);
        printf("Date of joining: %s\n", p.date_of_joining);
        printf("Salary: %.2f\n", p.salary);
    }
```

output:

```
Enter person name: Akshay

Enter date of joining (DD/MM/YYYY): 12/10/2018

Enter salary: 75000

Person name: Akshay

Date of joining: 12/10/2018

Salary: 75000.00
```

PRACTICAL 11: Design a structure student_records to contain Roll_no, Name, City and Percentage obtained.Develop a program to read data for 5 students and Display them.

```
#include <stdio.h>

struct student_records
{
    int Roll_no;
    char Name[50];
    char City[50];
    float Percentage;
};

void main()
{
    struct student_records students[5];

    // Read data for 5 students
    for (int i = 0; i < 5; i++)
    {
        printf("Enter details for student %d:\n", i + 1);
        printf("Roll no: ");
        scanf("%d", &students[i].Roll_no);
        printf("Name: ");
        scanf("%s", students[i].Name);
        printf("City: ");
        scanf("%s", students[i].City);
        printf("Percentage obtained: ");
        scanf("%f", &students[i].Percentage);
    }

    // Display the data for 5 students
    printf("\nDetails of 5 students:\n");
    for (int i = 0; i < 5; i++)
    {
        printf("Student %d:\n", i + 1);
        printf("Roll no: %d\n", students[i].Roll_no);
        printf("Name: %s\n", students[i].Name);
        printf("City: %s\n", students[i].City);
        printf("Percentage obtained: %.2f\n", students[i].Percentage);
        printf("\n");
    }
}
```

output:

```
Enter details for student 1:
Roll no: 01
Name: Bheem
City: IDAR
Percentage obtained: 89
Enter details for student 2:
Roll no: 02
Name: Raju
City: HMT
```

```
Percentage obtained: 80
Enter details for student 3:
Roll no: 03
Name: Jaggu
City: VADALI
Percentage obtained: 86
Enter details for student 4:
Roll no: 04
Name: Chutki
City: KHEDBRAHMA
Percentage obtained: 90
Enter details for student 5:
Roll no: 05
Name: Kaliya
City: DHOLAKPUR
Percentage obtained: 95

Details of 5 students:
Student 1:
Roll no: 1
Name: Bheem
City: IDAR
Percentage obtained: 89.00

Student 2:
Roll no: 2
Name: Raju
City: HMT
Percentage obtained: 80.00

Student 3:
Roll no: 3
Name: Jaggu
City: VADALI
Percentage obtained: 86.00

Student 4:
Roll no: 4
Name: Chutki
City: KHEDBRAHMA
Percentage obtained: 90.00

Student 5:
Roll no: 5
Name: Kaliya
City: DHOLAKPUR
Percentage obtained: 95.00
```

PRACTICAL 12: Write a program using structure within structure.

```
#include <stdio.h>

struct date {
    int day;
    int month;
    int year;
};

struct student {
    int roll_no;
    char name[50];
    struct date dob;
};

void main() {
    struct student s;

    printf("Enter student details:\n");
    printf("Roll no: ");
    scanf("%d", &s.roll_no);
    printf("Name: ");
    scanf("%s", s.name);
    printf("Date of birth (dd-mm-yyyy): ");
    scanf("%d-%d-%d", &s.dob.day, &s.dob.month, &s.dob.year);

    printf("\nStudent details:\n");
    printf("Roll no: %d\n", s.roll_no);
    printf("Name: %s\n", s.name);
    printf("Date of birth: %02d-%02d-%04d\n", s.dob.day, s.dob.month, s.dob.year);

}
```

output:

```
Enter student details:

Roll no: 01

Name: Krishna

Date of birth (dd-mm-yyyy): 01-01-2005


Student details:

Roll no: 1

Name: Krishna

Date of birth: 01-01-2005
```

PRACTICAL 13: Write a program using structure within Function.

```
#include <stdio.h>

struct date {
    int day;
    int month;
    int year;
};

struct student {
    int roll_no;
    char name[50];
    struct date dob;
};

void printStudent(struct student s) {
    printf("Roll no: %d\n", s.roll_no);
    printf("Name: %s\n", s.name);
    printf("Date of birth: %02d-%02d-%04d\n", s.dob.day, s.dob.month, s.dob.year);
}

void main() {
    struct student s;

    printf("Enter student details:\n");
    printf("Roll no: ");
    scanf("%d", &s.roll_no);
    printf("Name: ");
    scanf("%s", s.name);
    printf("Date of birth (dd-mm-yyyy): ");
    scanf("%d-%d-%d", &s.dob.day, &s.dob.month, &s.dob.year);

    printf("\nStudent details:\n");
    printStudent(s);

}
```

output:

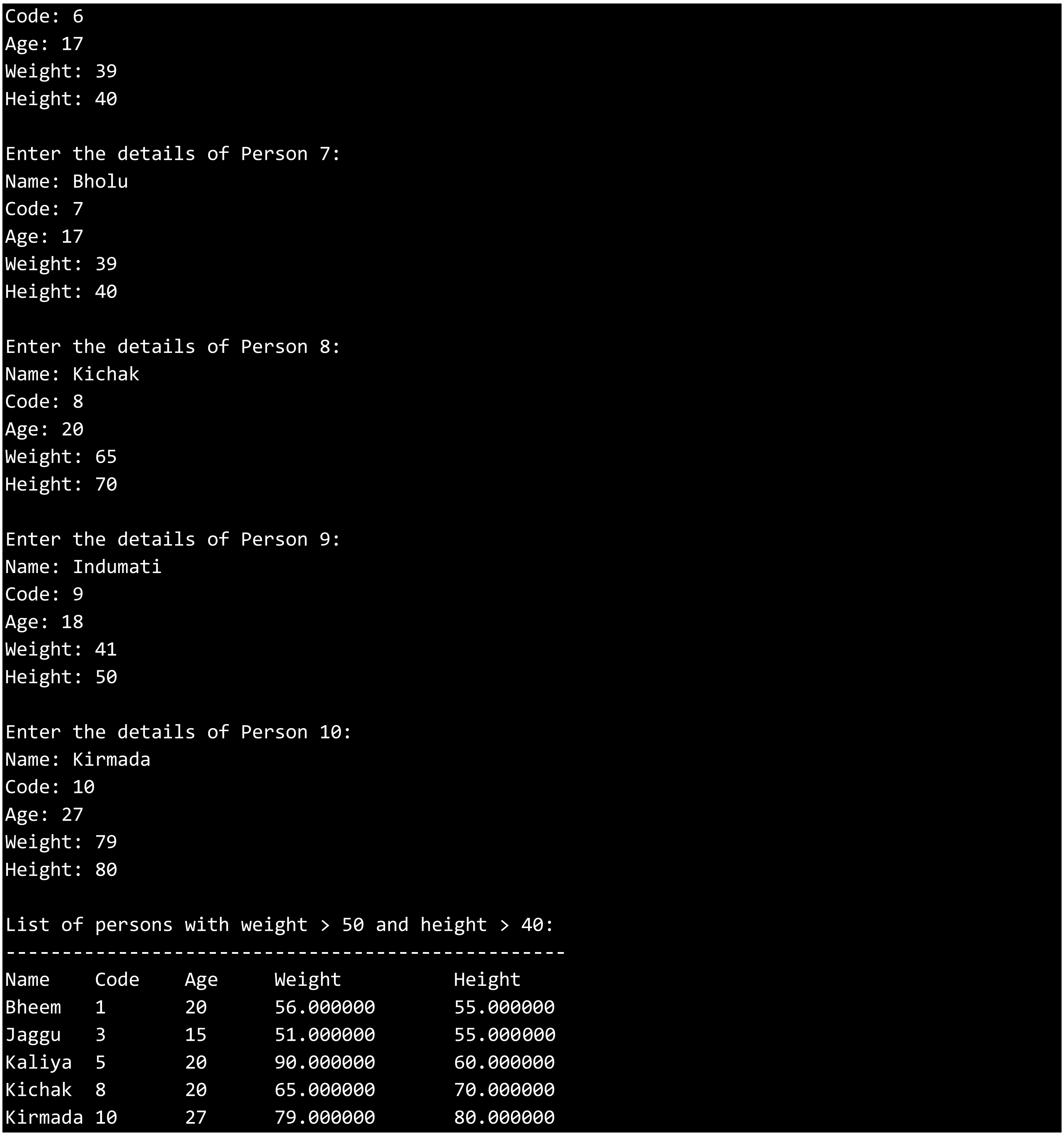
```
Enter student details:

Roll no: 001

Name: Arjun

Date of birth (dd-mm-yyyy): 03-01-2005



Student details:

Roll no: 1

Name: Arjun

Date of birth: 03-01-2005
```

PRACTICAL 14: Write a program declare following structure member: name, code, age, weight and height. Read all members of the structure for 10 persons and find list of persons with all related data whose weight > 50 and height > 40 and print the same with suitable format and title.

```c
#include <stdio.h>

#include <string.h>


struct person {

    char name[50];

    int code;

    int age;

    float weight;

    float height;

};


int main() {

    struct person p[10];

    int i;

    for (i = 0; i < 10; i++) {

        printf("\nEnter the details of Person %d:\n", i+1);

        printf("Name: ");

        scanf("%s",&p[i].name);

        printf("Code: ");

        scanf("%d", &p[i].code);

        printf("Age: ");

        scanf("%d", &p[i].age);

        printf("Weight: ");

        scanf("%f", &p[i].weight);

        printf("Height: ");

        scanf("%f", &p[i].height);

        getchar(); // consume newline character left by scanf

    }

    printf("\nList of persons with weight > 50 and height > 40:\n");

    printf("--------------------------------------------------\n");
    printf("%s\t%s\t%s\t%s\t\t%s\n", "Name", "Code", "Age", "Weight", "Height");

    for (i = 0; i < 10; i++) {

        if (p[i].weight > 50 && p[i].height > 40) {

            printf("%s\t%d\t%d\t%f\t%f\n", p[i].name, p[i].code, p[i].age, p[i].weight,
            p[i].height);

        }

    }

    return 0;

}
```

output:

```
Enter the details of Person 1:
Name: Bheem
Code: 1
Age: 20
Weight: 56
Height: 55

Enter the details of Person 2:
Name: Raju
Code: 2
Age: 18
Weight: 46
Height: 39

Enter the details of Person 3:
Name: Jaggu
Code: 3
Age: 15
Weight: 51
Height: 55

Enter the details of Person 4:
Name: Chutki
Code: 4
Age: 17
Weight: 47
Height: 50

Enter the details of Person 5:
Name: Kaliya
Code: 5
Age: 20
Weight: 90
Height: 60

Enter the details of Person 6:
Name: Dholu
```



```
Code: 6
Age: 17
Weight: 39
Height: 40

Enter the details of Person 7:
Name: Bholu
Code: 7
Age: 17
Weight: 39
Height: 40

Enter the details of Person 8:
Name: Kichak
Code: 8
Age: 20
Weight: 65
Height: 70

Enter the details of Person 9:
Name: Indumati
Code: 9
Age: 18
Weight: 41
Height: 50

Enter the details of Person 10:
Name: Kirmada
Code: 10
Age: 27
Weight: 79
Height: 80

List of persons with weight > 50 and height > 40:
---------------------------------------------------
Name    Code    Age     Weight          Height
Bheem   1       20      56.000000       55.000000
Jaggu   3       15      51.000000       55.000000
Kaliya  5       20      90.000000       60.000000
Kichak  8       20      65.000000       70.000000
Kirmada 10      27      79.000000       80.000000
```

PRACTICAL 15: Write a program to use of pointer in arithmetic operation.

```c
#include <stdio.h>

void main() {
    int a = 10, b = 5;
    int *ptr1, *ptr2;

    ptr1 = &a;
    ptr2 = &b;

    printf("a = %d, b = %d\n", a, b);
    printf("*ptr1 = %d, *ptr2 = %d\n", *ptr1, *ptr2);

    // pointer arithmetic
    int sum = *ptr1 + *ptr2;
    int diff = *ptr1 - *ptr2;
    int prod = (*ptr1) * (*ptr2);
    int quot = (*ptr1) / (*ptr2);

    printf("sum = %d, diff = %d, prod = %d, quot = %d\n", sum, diff, prod, quot);

}
```

output:

```
*ptr1 = 10, *ptr2 = 5

sum = 15, diff = 5, prod = 50, quot = 2
```

PRACTICAL 16: Write a program to accept 10 numbers and display its sum using pointer.

```
#include <stdio.h>

void main() {
    int num[10], sum = 0, *ptr;

    // reading input values
    printf("Enter 10 numbers:\n");
    for (int i = 0; i < 10; i++) {
        scanf("%d", &num[i]);
    }

    // summing the values using pointer
    ptr = num; // pointing to the first element of array
    for (int i = 0; i < 10; i++) {
        sum += *ptr;
        ptr++; // moving to next element of array
    }

    // displaying the sum
    printf("The sum of the 10 numbers is: %d\n", sum);

}
```

output:

```
Enter 10 numbers:

5

3

7

2

8

1

5

6

3

2

The sum of the 10 numbers is: 42
```

PRACTICAL 17: Write a program to accept 10 numbers and sort them with use of pointer.

```c
#include <stdio.h>

void main() {

    int num[10], temp, *ptr;

    // reading input values

    printf("Enter 10 numbers:\n");

    for (int i = 0; i < 10; i++) {

        scanf("%d", &num[i]);

    }

    // sorting the values using pointer

    ptr = num; // pointing to the first element of array

    for (int i = 0; i < 10; i++) {

        for (int j = i + 1; j < 10; j++) {

            if (*(ptr + j) < *(ptr + i)) {

                // swapping the elements using temporary variable

                temp = *(ptr + i);

                *(ptr + i) = *(ptr + j);

                *(ptr + j) = temp;

            }

        }

    }

    // displaying the sorted values

    printf("The sorted numbers are:\n");

    for (int i = 0; i < 10; i++) {

        printf("%d ", *(ptr + i));

    }

    printf("\n");

}
```

output:

```
Enter 10 numbers:

6

9

3
```

```
2

6

1

0

6

7

9

The sorted numbers are:

0 1 2 3 6 6 6 7 9 9
```

PRACTICAL 18: Write a program to swap the two values using pointers and UDF.

```
#include <stdio.h>

void swap(int *a, int *b) { // function to swap two values

    int temp = *a;

    *a = *b;

    *b = temp;

}

void main() {

    int num1, num2;

    printf("Enter two values: ");

    scanf("%d %d", &num1, &num2);

    printf("Before swapping: num1 = %d, num2 = %d\n", num1, num2);

    // swapping the values using function call and pointers

    swap(&num1, &num2);

    printf("After swapping: num1 = %d, num2 = %d\n", num1, num2);

}
```

output:

```
Enter two values: 15

20

Before swapping: num1 = 15, num2 = 20

After swapping: num1 = 20, num2 = 15
```

PRACTICAL 19: Write a program with structure and pointer.

```
#include <stdio.h>

struct student {

    char name[50];

    int roll_number;

    float marks;

};

void main() {

    struct student s1 = {"John Doe", 101, 78.5}; // initializing structure variable

    struct student *ptr = &s1; // declaring pointer to structure variable


    printf("Details of the student:\n");

    printf("Name: %s\n", ptr->name); // accessing structure member using pointer

    printf("Roll Number: %d\n", ptr->roll_number);

    printf("Marks: %.2f\n", ptr->marks);

}
```

output:

```
Details of the student:

Name: John Doe

Roll Number: 101

Marks: 78.50
```

PRACTICAL 20: Write a program using pointer to determine the length of a character string.

```
#include <stdio.h>
int string_length(char *str) { // function to calculate string length using pointer
    int len = 0;
    while (*str != '\0') { // loop until end of string is reached
        len++;
        str++;
    }
    return len;
}
void main() {
    char str[50];
    printf("Enter a string: ");
    scanf("%s", str);
    int len = string_length(str);
    printf("The length of the string is: %d\n", len);
}
```

output:

```
Enter a string: Hello

The length of the string is: 5
```

PRACTICAL 21: Write a program using pointers to read an array of integers and print its elements in reverse order.

```c
#include <stdio.h>
int main() {
    int arr[50], size;
    printf("Enter the size of the array: ");
    scanf("%d", &size);
    printf("Enter %d elements of the array:\n", size);
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }
    int *ptr = arr + size - 1; // initializing pointer to the last element of the array
    printf("The elements of the array in reverse order are:\n");
    while (ptr >= arr) { // loop until the first element of the array is reached
        printf("%d ", *ptr); // printing the element using pointer
        ptr--; // moving the pointer to the previous element
    }
}
```

output:

```
Enter the size of the array: 5
Enter 5 elements of the array:
1
2
3
4
5
The elements of the array in reverse order are:
5 4 3 2 1
```

PRACTICAL 22: Write a program using UDF and pointers to add two matrices and to return the resultant matrix to the calling function.

```
#include <stdio.h>

#include <stdlib.h>


int **add_matrices(int **mat1, int **mat2, int rows, int cols) {

    int **result = malloc(rows * sizeof(int *)); // allocate memory for the resultant matrix

    for (int i = 0; i < rows; i++) {

        result[i] = malloc(cols * sizeof(int));

        for (int j = 0; j < cols; j++) {

            result[i][j] = mat1[i][j] + mat2[i][j];

        }

    }

    return result;

}

int main() {

    int rows, cols;

    printf("Enter the number of rows and columns of the matrices: ");

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

    int **mat1 = malloc(rows * sizeof(int *)); // allocate memory for the first matrix

    printf("Enter the elements of the first matrix:\n");

    for (int i = 0; i < rows; i++) {

        mat1[i] = malloc(cols * sizeof(int));

        for (int j = 0; j < cols; j++) {

            scanf("%d", &mat1[i][j]);

        }

    }

    int **mat2 = malloc(rows * sizeof(int *)); // allocate memory for the second matrix

    printf("Enter the elements of the second matrix:\n");

    for (int i = 0; i < rows; i++) {

        mat2[i] = malloc(cols * sizeof(int));

        for (int j = 0; j < cols; j++) {

            scanf("%d", &mat2[i][j]);

        }
```

```c
    }
    int **result = add_matrices(mat1, mat2, rows, cols);
    printf("The resultant matrix is:\n");
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            printf("%d ", result[i][j]);
        }
        printf("\n");
    }
    // free the memory allocated for the matrices and the resultant matrix
    for (int i = 0; i < rows; i++) {
        free(mat1[i]);
        free(mat2[i]);
        free(result[i]);
    }
    free(mat1);
    free(mat2);
    free(result);
    return 0;
}
```

output:

```
Enter the number of rows and columns of the matrices: 2
2
Enter the elements of the first matrix:
1
2
3
4
Enter the elements of the second matrix:
5
6
7
8
The resultant matrix is:
6 8
```

```
10 12
```

PRACTICAL 23: Create one text file store some information into it and print the same information on Terminal.

```
#include <stdio.h>

int main() {

    FILE *fp;

    char filename[] = "info.txt";

    char name[10];

    int age;

    float height;

    char ch;

    // open the file for writing

    fp = fopen("info.txt", "w");

    // write information to the file

    printf("Enter name:");

    scanf("%s",&name);

    printf("Enter age:");

    scanf("%d",&age);

    printf("Enter height:");

    scanf("%f",&height);

    fprintf(fp, "Name: %s\nAge: %d\nHeight: %.2f",name,age,height);

    fclose(fp);

    // open the file for reading

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

    // read and print the information from the file

    printf("The information stored in the file is:\n");

    while (!feof(fp))

    {

        ch=getc(fp);

        printf("%c",ch);

    }

    fclose(fp);

    return 0;

}
```

output:

```
Enter name:Karna

Enter age:21

Enter height:7

The information stored in the file is:

Name: Karna

Age: 21

Height: 7.00
```

PRACTICAL 24: A file named data contains series of integer no. Write a c program to read that no. and then write all odd no into file named odd no. and write all even no into file named even no. Display all the contents of these file on screen.

```c
#include <stdio.h>

int main() {

   FILE *data_file, *odd_file, *even_file;

   int num;

   // Open the data file in read mode

   data_file = fopen("data.txt", "r");

   // Open the odd number file in write mode

   odd_file = fopen("odd_no.txt", "w");

   // Open the even number file in write mode

   even_file = fopen("even_no.txt", "w");

   // Read integers from the data file and write them to the appropriate file

   while(fscanf(data_file, "%d", &num) == 1) {

      if(num % 2 == 0) {

         fprintf(even_file, "%d\n", num);

      } else {

         fprintf(odd_file, "%d\n", num);

      }

   }

   // Close all the files

   fclose(data_file);

   fclose(odd_file);

   fclose(even_file);


   // Open the odd number file in read mode and display its contents on the screen

   odd_file = fopen("odd_no.txt", "r");

   printf("Odd Numbers:\n");

   while(fscanf(odd_file, "%d", &num) == 1) {

      printf("%d\n", num);

   }

   // Close the odd number file
```

```c
    fclose(odd_file);

    // Open the even number file in read mode and display its contents on the screen

    even_file = fopen("even_no.txt", "r");

    printf("Even Numbers:\n");

    while(fscanf(even_file, "%d", &num) == 1) {

        printf("%d\n", num);

    }

    // Close the even number file

    fclose(even_file);

    return 0;

}
```

output:

```
Odd Numbers:

1

3

5

7

9

11

Even Numbers:

2

4

6

8

10

12
```

PRACTICAL 25: Write a c program to read data from keyboard write it to a file called input and Display data of input file on the screen.

```
#include <stdio.h>
int main()
{
    FILE *input; // creating file pointer
    // open file in write mode
    input = fopen("input", "w");
    // check if the file created successfully
    if (input == NULL)
    {
        printf("Error opening file!\n");
        return 0;
    }
    char c; // creating character variable
    // prompting user to enter age
    printf("Please enter your age: ");
    // looping until user enter Enter or G
    while ((c = getchar()) != '\n' && c != EOF)
    {
        putc(c, input); // writing in file
    }
    // closing file
    fclose(input);
    // open file again in read mode
    input = fopen("input", "r");
    // print the contents of file
    printf("\nYour Details:\n");
    while ((c = fgetc(input)) != EOF)
    {
        printf("%c", c);
    }
    printf("\n");
    // close opened file
    fclose(input);

    return 0;

}
```

output:

```
Please enter your age: 18


Your Details:

18
```

PRACTICAL 26: Write a c program to read data from keyboard write it to a file called input and Display data of input file on the screen.

```c
#include <stdio.h>

#include <stdlib.h>

void main()

{

    int characterCount = 0, lineCount= 0;

    char c;

    FILE *fp;

    fp = fopen("pr26.txt","r");

    if (fp == NULL)

    {

        printf("File not found! \n");

    }

    while ((c = fgetc(fp)) != EOF)

    {

        characterCount++;

        if (c == '\n')

        {

            lineCount++;

        }

    }

    printf("Number of characters = %d \n", characterCount);

    printf("Number of lines = %d \n", lineCount);

    fclose(fp);

}
```

output:

```
Number of characters = 30

Number of lines = 1
```

PRACTICAL 27: Write a c program to read mark data which contains roll no, name, sub1, sub2, sub3 file and generate the annual examination results are tabulated as follows

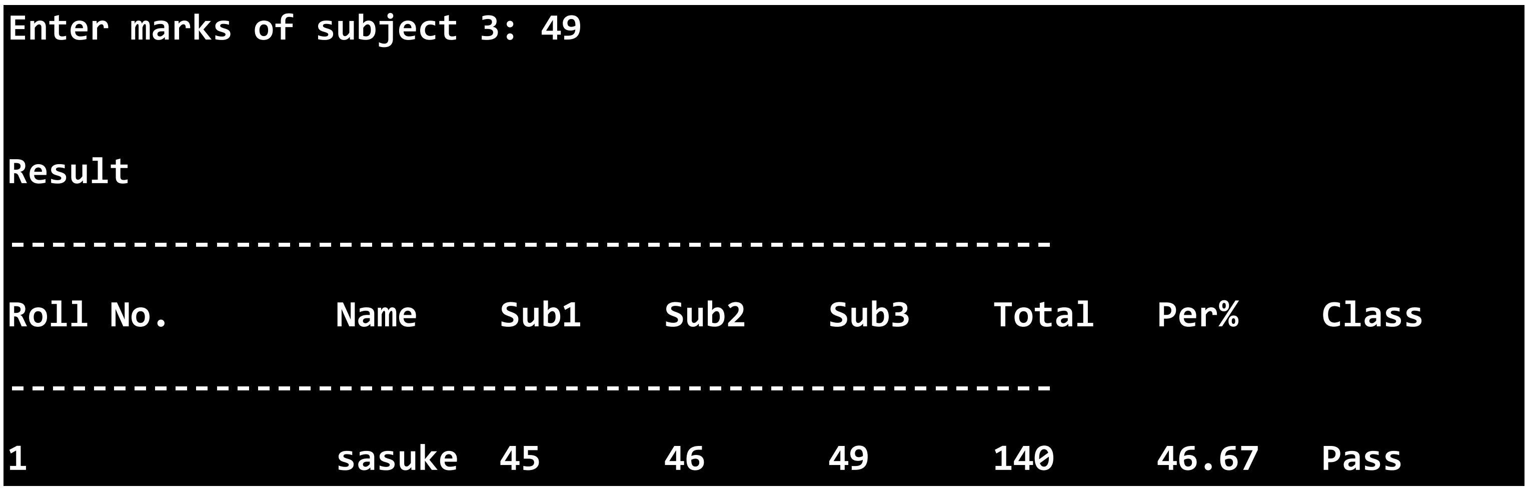
---

Roll no Name Sub1 Sub2 Sub3 Total per % Class

```
#include <stdio.h>

void main()
{
    int rollNo, sub1, sub2, sub3, total;
    float percentage;
    char name[10];
    char class[5];

    // Read the roll number, name, and marks of the student
    printf("Enter roll number: ");
    scanf("%d", &rollNo);

    printf("Enter name: ");
    scanf("%s", &name);

    printf("Enter marks of subject 1: ");
    scanf("%d", &sub1);

    printf("Enter marks of subject 2: ");
    scanf("%d", &sub2);

    printf("Enter marks of subject 3: ");
    scanf("%d", &sub3);

    // Calculate total and percentage
    total = sub1 + sub2 + sub3;
    percentage = (float)total / 3;

    // Determine the class based on percentage
    if (percentage >= 60)
    {
        sprintf(class, "First");
    }
    else if (percentage >= 50)
    {
        sprintf(class, "Second");
    }
    else if (percentage >= 40)
    {
        sprintf(class, "Pass");
    }
    else
    {
        sprintf(class, "Fail");
    }
    // Display the result
    printf("\nResult\n");
    printf("---------------------------------------------------\n");
    printf("Roll No.\tName\tSub1\tSub2\tSub3\tTotal\tPer%%\tClass\n");
    printf("---------------------------------------------------\n");
    printf("%d\t\t%s\t%d\t%d\t%d\t%d\t%.2f\t%s\n", rollNo, name, sub1, sub2, sub3, total,
percentage, class);
}
```

output:

```
Enter roll number: 1

Enter name: sasuke

Enter marks of subject 1: 45

Enter marks of subject 2: 46
```



```
Enter marks of subject 3: 49


Result

---------------------------------------------------

Roll No.        Name    Sub1    Sub2    Sub3    Total   Per%    Class

---------------------------------------------------

1               sasuke  45      46      49      140     46.67   Pass
```

PRACTICAL 28: Write a c program to input employee no, employee name and basic and to store output into empdata file in following format.

```
#include <stdio.h>
int main()
{
    int empNo, basic;
    float da, hra, ma, pf, gross, netPay;
    char name[30];
    // Read the employee number, name, and basic salary
    printf("Enter employee number: ");
    scanf("%d", &empNo);

    printf("Enter name: ");
    scanf("%s", name);

    printf("Enter basic salary: ");
    scanf("%d", &basic);

    // Calculate DA, HRA, MA, PF, Gross, and Net Pay
    da = basic * 0.5;
    hra = basic * 0.1;
    ma = 100;
    pf = basic * 0.1;
    gross = basic + da + hra + ma;
    netPay = gross - pf;

    // Open the empdata file for writing
    FILE *fp = fopen("empdata.txt", "a");
    if (fp == NULL)
    {
        printf("Error opening file!\n");
        return 1;
    }
    // Write the employee data to the file

    fprintf(fp, "%d %s %d %.2f %.2f %.2f %.2f %.2f %.2f\n", empNo, name, basic, da, hra, ma, 
pf, gross, netPay);


    // Close the file

    fclose(fp);


    // Display the result

    printf("\nA/c Department\n");

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

    printf("Emp-No  Name  Basic  DA  HRA  MA  PF  GROSS  NET-PAY\n");

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

    printf("%d  %s  %d  %.2f  %.2f  %.2f  %.2f  %.2f  %.2f\n", empNo, name, basic, da, hra, 
ma, pf, gross, netPay);


    return 0;

}
```

output:

```
Enter employee number: 102

Enter name: Haresh

Enter basic salary: 25600


A/c Department

---------------------------------------------------

Emp-No  Name  Basic  DA  HRA  MA  PF  GROSS  NET-PAY

---------------------------------------------------

102  Haresh  25600  12800.00  2560.00  100.00  2560.00  41060.00  38500.00
```

PRACTICAL 29: Write a c program to read empin data file which contains empno, empname and basic. To create empout data file as per practical no 33 format.

```
#include <stdio.h>

#include <stdlib.h>


int main{

    int empno;

    char empname[30];

    int basic;

    FILE *empfile;

    empfile = fopen("empin.txt", "r");

    if (empfile == NULL)

    {

        printf("Error reading file!\n");

        return 1; //exit the program if file opening fails

    }

    FILE *empoutfile = fopen("empout.txt", "a"); // Create output file

    if (empoutfile == NULL) // Check if output file was created successfully

    {

        printf("Error creating file!\n");

        return 1; //exit the program if file opening fails

    }

    while (fscanf(empfile, "%d %s %d", &empno, empname, &basic) == 3)

    {

        int total = basic + (basic * 0.3);

        fprintf(empoutfile, "Empno: %d\nName: %s\nBasic: %d\nTotal: %d\n", empno, empname,
        basic, total);

    }

    fclose(empfile);

    fclose(empoutfile);

    return 0;

}
```

output:

empout.txt

```
Empno: 101

Name: Haresh

Basic: 20500

Total: 26649
```

PRACTICAL 30:Write a program using fseek and ftell functions.

```
#include <stdio.h>

int main()

{

    /* Declaring new variables */

    FILE *fp;

    char ch;

    /* Opening a file "example.txt" in read mode */

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

    /* Checking whether it is empty or not */

    if (fp == NULL)

    {

        printf("File is empty\n");

    }

    /* Navigate to the fifth character in the file */

    fseek(fp, 5, SEEK_SET);

    /* Get position of current character */

    ftell(fp);

    /* Read the character at current location */

    ch = fgetc(fp);

    printf("The character at offset 5 is %c\n", ch);

    /* Closing the file */

    fclose(fp);

}
```

output:

```
    The character at offset 5 is W
```

PRACTICAL 31: Two files DATA1 and DATA2 contain sorted lists of integers. Write a program to produce a third file DATA which holds a single sorted, merged list of these two lists.Use command line arguments to specify the file names.

```
#include <stdio.h>

#include <stdlib.h>


int main( int arg[BC[BCA, Cyber Security]yber Security]har *argv[]) {

  FILE *fp1; //creates a file pointer

  FILE *fp2; //creates a file pointer

  FILE *fp3; //creates a file pointer

  int n1, n2; //variable to store integer numbers

  fp1=fopen(argv[1],"r"); //command to open DATA1 text file in read mode

  fp2=fopen(argv[2],"r"); //command to open DATA2 text file in read mode

  fp3=fopen("DATA","w"); //command to open a new file DATA in write mode

  // loop to read numbers from text files and compare them

  while(fscanf(fp1,"%d",&n1) != EOF && fscanf(fp2,"%d",&n2) != EOF) {

    if (n1 <= n2) {

      fprintf(fp3,"%d \n",n1); //write the smaller number into DATA

      fscanf(fp1,"%d",&n1);    //update n1 so that we can compare the next number

    } else {

      fprintf(fp3,"%d \n",n2); //write the smaller number into DATA

      fscanf(fp2,"%d",&n2);    //update n2 so that we can compare the next numbers

    }

  }

 // to write the leftover numbers from either file

  while(fscanf(fp1,"%d",&n1) != EOF)

    fprintf(fp3,"%d \n",n1);

  while(fscanf(fp2,"%d",&n2) != EOF)

    fprintf(fp3,"%d \n",n2);

  fclose(fp1); // close the DATA1 text file

  fclose(fp2); // close the DATA2 text file

  return 0;

}
```

output:

```
Enter the name of the first file: DATA1
Enter the name of the second file: DATA2
The merged file is: DATA
```

PRACTICAL 32: Write a C program to work as a dos type command using command line argument.

```c
#include <stdio.h>

#include <string.h>

int main(int argc, char *argv[])

{

    int i;

    // Check if there are any command line arguments passed

    if (argc > 1)

    {

        // Check the first command line argument

        if (strcmp(argv[1], "find_file") == 0)

        {

            // Execute the find_file dos command

            printf("\n Executing the find_file DOS Command\n");

            printf("\n Finding a file...");

        }

        else

        {

            printf("\n Error: Unknown command \n");

        }

    }

    else

    {

        printf("\n Error: Please provide a command to execute \n");

    }

}
```

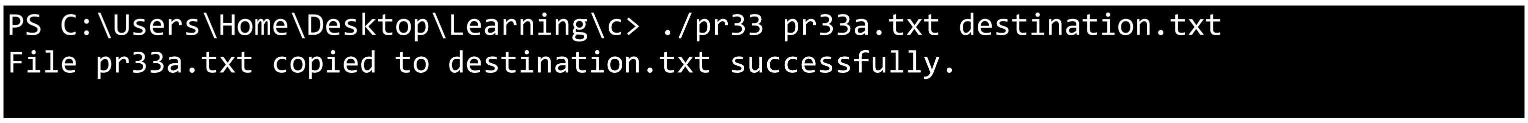
output:

```
PS C:\Users\Home\Desktop\Learning\c> gcc pr32.c -o pr32
PS C:\Users\Home\Desktop\Learning\c> ./pr32 find_file

 Executing the find_file DOS Command

 Finding a file...
```

PRACTICAL 33: Write a C program to work as a dos copy command using command line argument.

```c
#include <stdio.h>

#include <stdlib.h>


int main(int argc, char *argv[]) {

    FILE *source_file, *destination_file;

    char c;

    if (argc != 3) {

        printf("Usage: %s source_file destination_file\n", argv[0]);

        return 1;

    }

    source_file = fopen(argv[1], "r");

    if (source_file == NULL) {

        printf("Error: Cannot open source file %s\n", argv[1]);

        return 1;

    }

    destination_file = fopen(argv[2], "w");

    if (destination_file == NULL) {

        printf("Error: Cannot open destination file %s\n", argv[2]);

        return 1;

    }

    while ((c = fgetc(source_file)) != EOF) {

        fputc(c, destination_file);

    }

    printf("File %s copied to %s successfully.\n", argv[1], argv[2]);

    fclose(source_file);

    fclose(destination_file);


    return 0;

}
```

output:



```
PS C:\Users\Home\Desktop\Learning\c> ./pr33 pr33a.txt destination.txt
File pr33a.txt copied to destination.txt successfully.
```

PRACTICAL 34: Write programs which explain the use of memory allocation functions.

```c
#include <stdio.h>

#include <stdlib.h>

void main()

{

    // constant variable

    const int SIZE = 5;


    // memory allocation using malloc

    int *ptr = (int *)malloc(SIZE * sizeof(int));


    // using address of operator to store values

    *(ptr) = 10;

    *(ptr + 1) = 20;

    *(ptr + 2) = 30;

    *(ptr + 3) = 40;

    *(ptr + 4) = 50;


    // printing the values of ptr vector

    for (int i = 0; i < SIZE; i++)

        printf("%d ", *(ptr + i));


    // memory deallocation using free()

    free(ptr);

}
```

output:

```
10 20 30 40 50
```

PRACTICAL 35: Write a program which explains the use of macro.

```
#include <stdio.h>
// Define a macro with the name "N" and assign it a value of 8
#define N 8

int main(void) {
    // Declare a variable to store a value
    int value;

    /* Use the value of N stored in the macro
        to set the value of the variable "value" */
    value = N;

    // Print the value of the variable
    printf("The value is: %d", value);
}
```

output:

```
The value is: 8
```

* * * * *