|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 4 | Name of the Practical | Page NO. | Issue date | Submission date | Grade | Signature |
| 01. | Write a simple C program.   1. Print your name and address. 2. Find simple and compound interest | 4-5 | 07/08/2024 |  |  |  |
| 02. | Write a C program to swap two variables’ using  (i) third variable and (ii) without using a third variable. | 6 -7 | 08/08/2024 |  |  |  |
| 03. | Write a program to convert a given number of days into months and days using integer arithmetic operators. | 8 | 09/08/2024 |  |  |  |
| 04. | Write a program the use of variables in expression and their evaluation. | 9 | 14/08/2024 |  |  |  |
| 05. | Write a program converts the given temperature in Fahrenheit to Celsius using preprocessor. | 10 | 21/08/2024 |  |  |  |
| 06. | Write a program converts the given temperature in Fahrenheit to Celsius using preprocessor. | 11 | 22/08/2024 |  |  |  |
| 07. | Write a program to perform following tasks.   1. Find factorial of a number | 12-  13 | 23/08/2024 |  |  |  |
|  | b. Print prime numbers up N times. |  |  |
|  | Write a program to prepare the total |  |  |  |  |  |
| 08. | marks for N students by reading the  Reg. No, Name, Mark1 to Mark 6 by | 14- | 28/08/2024 |
|  | using array of structures. | 15 |  |
| 09. | Write a program using the function power (a, b) to calculate the value of a raised to b. | 16 | 29/08/2024 |  |  |  |
| 10. | Write a program to find the length of the given string using pointers. | 17 | 30/08/2024 |  |  |  |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| S.  NO. | Name of the Practical | Page NO. | Issue date | Submission date | Grade | Signature |
| 01. | Read an integer number, find the number of digit and sum of all individual digits and also print the above number  in reverse order. | 18 | 26/09/2024 |  |  |  |
| 02. | Write a program to perform following tasks   1. Print Fibonacci series up to N terms and its sum. 2. Print whether a given year is leap or not. | 19 | 27/09/2024 |  |  |  |
| 03. | Read a sentence through command line argument. Write a program to write out the string arguments to main in reverse order. | 20 | 03/10/2024 |  |  |  |
| 04. | Write a program to arrange the given N names in alphabetical order. | 21 | 04/10/2024 |  |  |  |
| 05. | Write a program to count the  numbers and chars in the string. | 22 | 09/10/2024 |  |  |  |
| 06. | Write a program that uses a function to sort an array of  integers. | 23 |  |  |  |  |
| 07. | Write a program to calculate the subject wise and student wise totals and store them as  a part of the structure. | 24-25 |  |  |  |  |
| 08. | Write a program to read 10 values to an array variable. Use pointers to locate and  display each value. | 26 |  |  |  |  |
| 09. | Write a program that uses a table of integers whose size will be specified  interactively at runtime. | 27 |  |  |  |  |
| 10. | Write a program to store a character string in a block of memory space created by MALLOC and then modify the same to store a larger  string. | 28 |  |  |  |  |

# Algorithm:

An algorithm is a step-by-step, well-defined procedure or set of rules used to solve a problem or perform a specific task. It takes input, processes it in a series of logical steps, and produces an output. Algorithms are fundamental in computer science and mathematics to solve problems efficiently and systematically.

### Key characteristics of an algorithm:

1. **Input:** Takes zero or more inputs.
2. **Output:** Produces at least one output.
3. **Definiteness:** Each step is clearly defined.
4. **Finiteness:** The algorithm terminates after a finite number of steps.
5. **Effectiveness:** Each step is basic enough to be performed practically.
   * **The algorithm and flowchart include the following three types of control structures:**
6. **Sequence**: In the sequence structure, statements are placed one after the other and the execution takes place starting from up to down.
7. **Branching (Selection)**: In branch control, there is a condition, and according to the condition, a decision of either TRUE or FALSE is achieved. In the case of TRUE, one of the two branches is explored, but in the case of FALSE condition, the other alternative is taken. Generally, the ‘IF-THEN’ is used to represent branch control.
8. **Loop (Repetition):** The Loop or Repetition allows a statement(s) to be executed repeatedly based on a certain loop condition, e.g., WHILE, FOR loops.

### Advantages of algorithms:

1. **Improved Problem-Solving:** Algorithms provide a step-by-step method for solving a problem, making it easier to break down complex problems into smaller, manageable parts.
2. **Efficiency:** Algorithms are designed to solve problems efficiently in terms of time and space. They help optimize the use of resources, such as memory and processing power.
3. **Clarity:** Algorithms are written in a structured and clear way, allowing anyone who understands the logic to follow and implement the steps.
4. **Reusability:** Once an algorithm is designed, it can be reused for solving similar types of problems without starting from scratch.
5. **Automation:** Algorithms allow for the automation of processes. A well-designed algorithm can be coded into a computer program, making tasks automated and reducing the need for human intervention.
6. **Debugging and Testing:** Since algorithms follow a logical sequence of steps, it is easier to trace errors or mistakes and debug them systematically.
7. **Scalability:** Algorithms can be adapted or modified to handle larger datasets or more complex scenarios as the problem size grows, making them scalable solutions.
8. **Consistency**: Algorithms ensure consistency in solving problems since the same steps will always produce the same results if given the same input.
   * **HOW TO WRITE ALGORITHMS**

**step 1: Define your algorithm's input:**

Many algorithms take in data to be processed, e.g., to calculate the area of a rectangle, input may be the rectangle's height and width.

**Step 2: Define the variables:**

Algorithm's variables allow you to use it in more than one place. We can define two variables for rectangle height and rectangle width as HEIGHT and WIDTH (or H & W). We should use meaningful variable names, e.g., instead of using H & W, use HEIGHT and WIDTH as variable names.

**Step 3: Outline the algorithm's operations:**

Use input variables for computation purposes. For example, to find the area of a rectangle, multiply the HEIGHT and WIDTH variables and store the value in a new variable (say) AREA. An algorithm's operations can take the form of multiple steps and even branch, depending on the value of the input variables.

**Step 4: Output the results of your algorithm's operations:**

In the case of the area of a rectangle, the output will be the value stored in the variable AREA. If the input variables described a rectangle with a HEIGHT of 2 and a WIDTH of 3, the algorithm would output the value of 6.

# FLOWCHART:

A flowchart is a visual representation of a process, algorithm, or workflow. It uses standardized symbols and shapes to illustrate the sequence of steps involved in completing a task or solving a problem. Flowcharts help in breaking down complex processes into simpler, easily understandable components, making them an essential tool for planning, analyzing, and communicating ideas.

### Advantages of flowchart:

* Flowchart is an excellent way of communicating the logic of a program.
* Easy and efficient to analyze problems using a flowchart.
* During the program development cycle, the flowchart plays the role of a blueprint, which makes the program development process easier.
* After successful development of a program, it needs continuous and timely maintenance during the course of its operation. The flowchart makes program or system maintenance easier.
* It is easy to convert the flowchart into any programming language code.

## Date:

# PART-A

**Practical- 01**

#### AIM: Write a simple C program. a. Print your name and address. b. Find simple and compound interest

1. Print your name and Address.

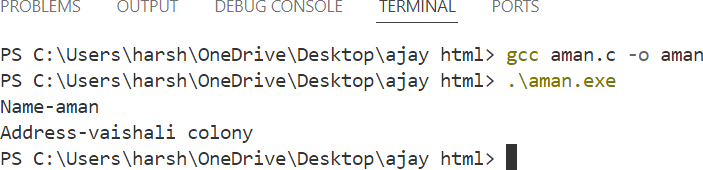
#include <stdio.h> int main()

{

**Output:**

printf("Name-Aman"); printf("Address-Vaishali colony");

}



#include<stdio.h> #include<string.h> int main()

{

char name[100]; char address[200];

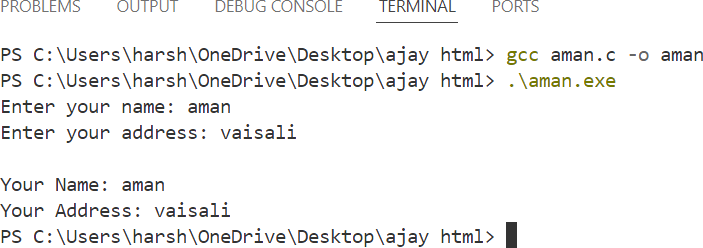
printf("Enter your name: "); scanf("%s", name); printf("Enter your address: "); scanf(" %s", address);

printf("\nYour Name: %s\n", name); printf("Your Address: %s\n", address);

return 0;

}

**Output:**

****

1. Find simple interest and compound interest

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

int main()

{

**Output:**

float principal, rate, time; printf("Enter principal amount: "); scanf("%f", &principal);

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

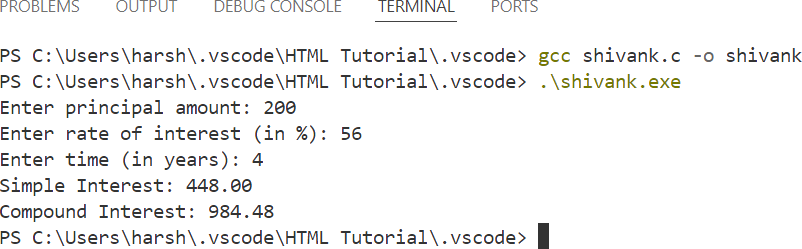
printf("Enter time (in years): "); scanf("%f", &time);

float simpleInterest = (principal \* rate \* time) / 100;

float compoundInterest = principal \* pow((1 + rate / 100), time) - principal; printf("Simple Interest: %.2f\n", simpleInterest);

printf("Compound Interest: %.2f\n", compoundInterest); return 0;

}



**Date:**

# Practical- 02

#### Aim: - W.A.P in c to swap variable’s using. a) Third variable b) Without using third variable

1. Third variable

#include<stdio.h> #include<conio.h> void main()

{

**Output:**

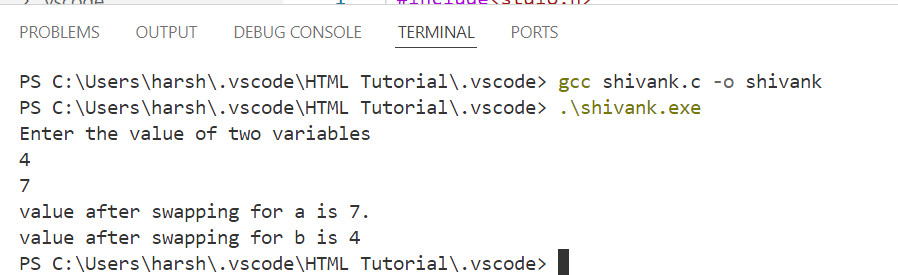
int a,b,c;

printf("Enter the value of two variables\n"); scanf("%d %d",&a,&b);

c=a; a=b; b=c;

printf("value after swapping for a is %d.\n",a); printf("value after swapping for b is %d",b);

}



1. Without using third variable

#include<stdio.h>

void main()

{

**Output:**

int a,b;

printf("Enter the first value:"); scanf("%d",&a);

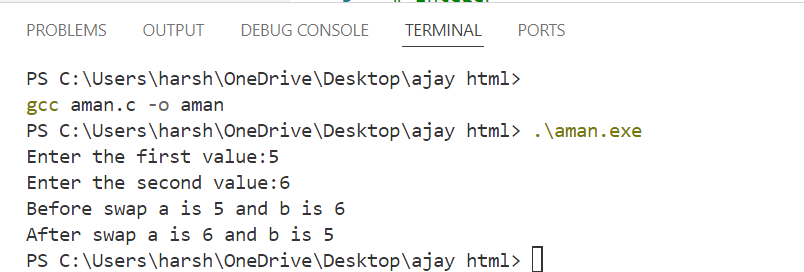
printf("Enter the second value:"); scanf("%d",&b);

printf("Before swap a is %d and b is %d\n",a,b); a=a+b;

b=a-b;

a=a-b;

printf("After swap a is %d and b is %d",a,b);



Aim :- W.A.P in c to convert a given number of days into months and days using integer arithmetic operations.

#include<stdio.h> #include<conio.h> void main()

{

**Output:**

int a,m,d;

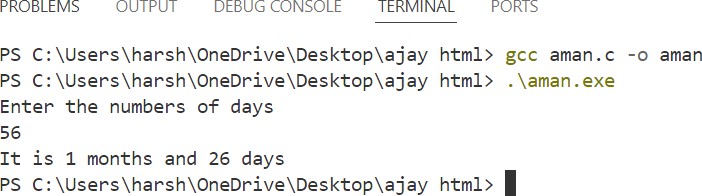
printf("Enter the numbers of days\n"); scanf("%d",&a);

m = a/30;

d = a– m\*30;

printf("It is %d months and %d days",m,d);

}



#### Aim :- W.A.P that use of variables in expression and their.

#include <stdio.h> void main()

{

int a, b, sum, difference, Product, Division;

printf("Enter value for a: "); scanf("%d", &a);

printf("Enter value for b: "); scanf("%d", &b);

sum = a + b; difference = a - b; product = a \* b; division = a / b;

printf("Sum: %d", sum); printf("\nDifference: %d", difference); printf("\nProduct: %d", product); printf("\nDivision: %d", division);

}

**Output:**

****

**Aim :- W.A.P in c that converts the given temperature in Fahrenheit to Celsius.**

#include <stdio.h> int main()

{

float Fahrenheit, Celsius;

printf("Enter the temperature in Fahrenheit: "); scanf("%f", &Fahrenheit);

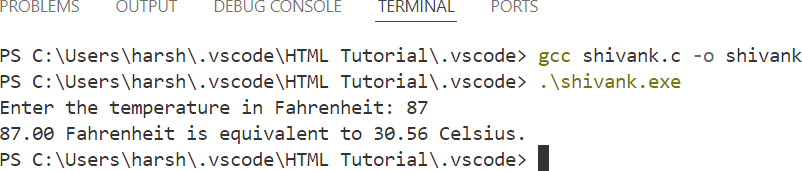
Celsius = ((Fahrenheit - 32) \* 5 / 9);

printf("%.2f Fahrenheit is equivalent to %.2f Celsius.\n", Fahrenheit, Celsius);

return 0;

}

**Output:**

****

#### Aim :- W.A.P in c to find the largest number between given three numbers.

#include <stdio.h> int main()

{

**Output:**

int a,b,c;

printf("Enter the three number\n"); scanf("%d%d%d",&a,&b,&c); if(a>b&&a>c){

printf("%d is the largest number",a,b,c);

}

else if(b>a&&b>c)

{

printf("%d is the largest number",b,a,c);

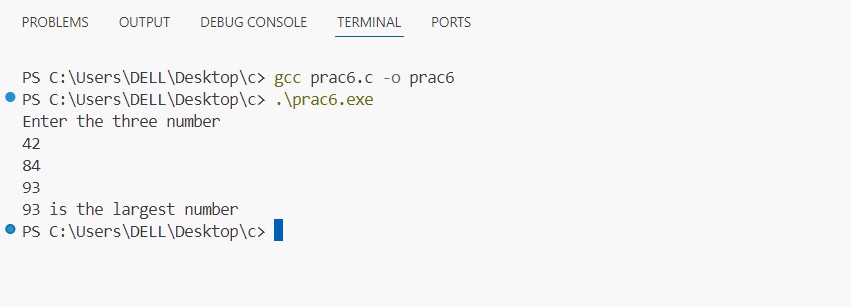
}

else{

printf("%d is the largest number",c,a,b);

}

}



# Practical- 07

#### Aim :- W.A.P to perform following tasks. a) Find factorial of a number. b) Print prime numbers up N times.

#include <stdio.h> int main()

{

**Output:**

int n,fact=1,i;

printf("Enter a Factorial number: "); scanf("%d", &n);

for(int i=1;i<=n;i++)

{

fact=fact\*i;

}

printf("%d",fact); return 0;

}



# PRACTICAL-08

**AIM:** Write a program to prepare the total marks for N students by reading the Reg. No, Name, Mark1 to Mark 6 by using array of structures.

#include <stdio.h> struct Student

{

int regNo;

char name[50]; float marks[6]; float totalMarks;

};

int main()

{

int N;

printf("Enter the number of students: "); scanf("%d", &N);

struct Student students[N]; for (int i = 0; i < N; i++) {

printf("\nEnter details for student %d:\n", i + 1);

printf("Registration Number: "); scanf("%d", &students[i].regNo);

printf("Name: ");

scanf(" %s\*%s", students[i].name); printf("Enter marks for 6 subjects:\n"); students[i].totalMarks = 0;

for (int j = 0; j < 6; j++) { printf("Mark %d: ", j + 1); scanf("%f", &students[i].marks[j]);

students[i].totalMarks += students[i].marks[j];

}

}

printf("\nTotal marks for each student:\n"); for (int i = 0; i < N; i++) {

printf("Student %d:\n", i + 1);

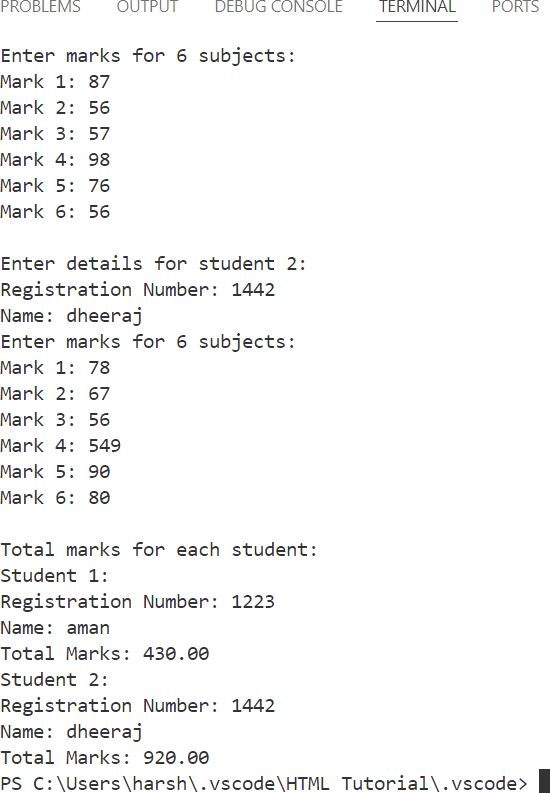
printf("Registration Number: %d\n", students[i].regNo); printf("Name: %s\n", students[i].name);

printf("Total Marks: %.2f\n", students[i].totalMarks);

}

return 0;}

**Output:**

****

#### Aim :- W.A.P using the function powr(a, b) to calculate the value of a raised to b.

#include<stdio.h> int aman(int a,int b)

{

int c; c=pow(a,b); return c;

}

int main()

{

int a,b,c;

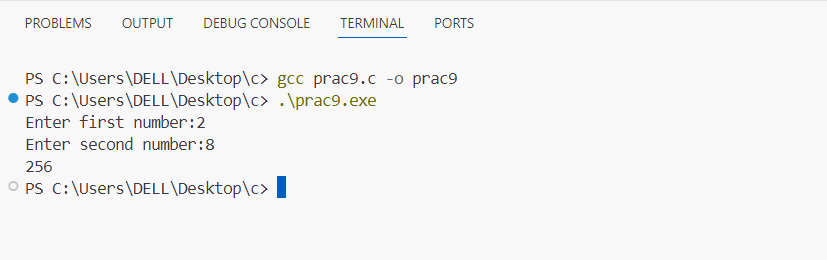
printf("Enter first number"); scanf("%d",&a);

printf("Enter second number"); scanf("%d",&b);

c=aman(a,b); printf("%d",c);

}

**Output:**

****

# Practical- 10

#### Aim :- W.A.P to find the length of the given string using pointers.

#include<stdio.h> #include<String.h> int main()

{

char aman[25]="Doing Practical of C"; int length=strlen(aman);

printf("%s”, aman); printf("%d",length); return 0;

}

**Output:**

****

# PART-B

**Practical- 01**

**AIM:** Read an integer number, find the number of digit and sum of all individual digits and also print the above number in reverse order.

#include <stdio.h>

int main() {

int number, originalNumber, digit, sum = 0, count = 0, reverse =0; printf("Enter an integer number: ");

scanf("%d", &number); if (number < 0)

{

number = -number;

}

originalNumber = number; while (number != 0)

{

digit = number % 10; sum += digit;

reverse = reverse \* 10 + digit; number /= 10;

count++;

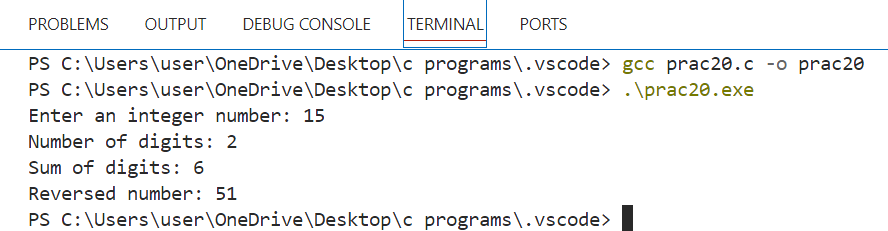
}

printf("Number of digits: %d\n", count); printf("Sum of digits: %d\n", sum); printf("Reversed number: %d\n", reverse);

return 0;

}

**Output:**

****

# Practical- 02

**AIM:** Write a program to perform following tasks a. Print Fibonacci series up to N terms and its sum. b. Print whether a given year is leap or not.

a. Print Fibonacci series up to N terms and its sum.

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

int N, t1 = 0, t2 = 1, nextTerm, sum = 0;

printf("Enter the number of terms in the Fibonacci series: "); scanf("%d", &N);

printf("Fibonacci Series: "); for (int i = 1; i <= N; i++) {

if (i == 1) {

printf("%d, ", t1); sum += t1; continue;

}

if (i == 2)

{

printf("%d, ", t2); sum += t2; continue;

}

nextTerm = t1 + t2; printf("%d, ", nextTerm); sum += nextTerm;

t1 = t2;

t2 = nextTerm;

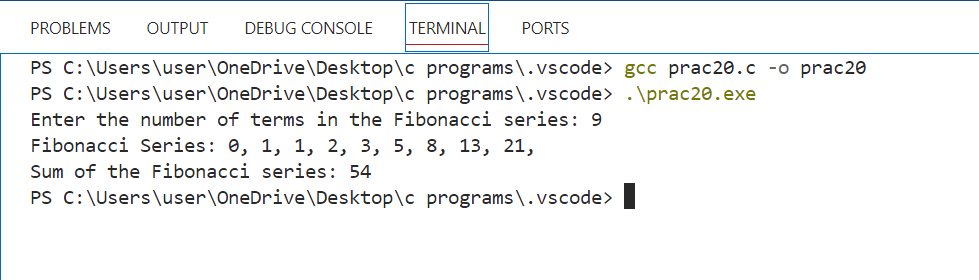
}

printf("\nSum of the Fibonacci series: %d\n", sum);

**Output:**

return 0;

}



**AIM:** Read a sentence through command line argument. Write a program to write out the string arguments to main in reverse order.

#include <stdio.h>

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

{

if (argc < 2)

{

printf(("usage: %s\"sentence\"\n",argv[0]);

return 1;

}

printf("Arguments in reverse order:\n"); for (int i = argc - 1; i > 0; i--) {

printf("%s ", argv[i]);

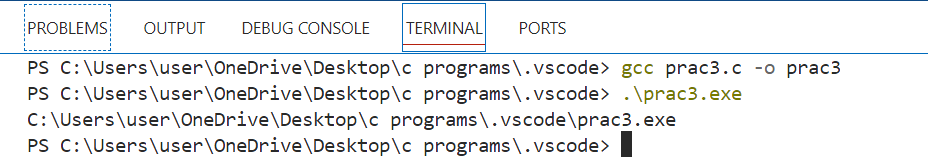
}

printf("\n");

return 0;

}

**Output:**

****

**AIM:** Write a program to arrange the given N names in alphabetical order.

#include <stdio.h> #include <string.h>

#define MAX\_NAMES 100

#define MAX\_LENGTH 50 int main() {

int N;

char names[MAX\_NAMES][MAX\_LENGTH]; printf("Enter the number of names: "); scanf("%d", &N);

printf("Enter the names:\n"); for (int i = 0; i < N; i++) {

printf("Name %d: ", i + 1); scanf(" %[^\n]%\*c", names[i]);

}

for (int i = 0; i < N - 1; i++) {

for (int j = 0; j < N - i - 1; j++) {

if (strcmp(names[j], names[j + 1]) > 0) { char temp[MAX\_LENGTH]; strcpy(temp, names[j]); strcpy(names[j], names[j + 1]); strcpy(names[j + 1], temp);

}

}

}

printf("\nNames in alphabetical order:\n"); for (int i = 0; i < N; i++) {

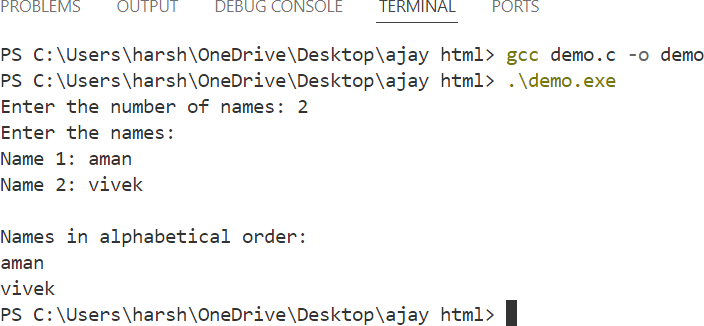
printf("%s\n", names[i]);

}

return 0;

}

**Output:**

****

**AIM:** Write a program to count the numbers and chars in the string.

#include <stdio.h> #include <ctype.h>

int count\_chars\_and\_numbers(const char \*input\_string, int \*char\_count, int \*digit\_count, int

\*other\_count) {

\*char\_count = 0;

\*digit\_count = 0;

\*other\_count = 0;

for (int i = 0; input\_string[i] != '\0'; i++) { if (isalpha(input\_string[i])) {

(\*char\_count)++;

} else if (isdigit(input\_string[i])) { (\*digit\_count)++;

} else {

(\*other\_count)++;

}

}

}

int main() {

char input\_string[100];

printf("Enter a string: ");

fgets(input\_string, sizeof(input\_string), stdin); // Use fgets to read the string int letters, digits, others;

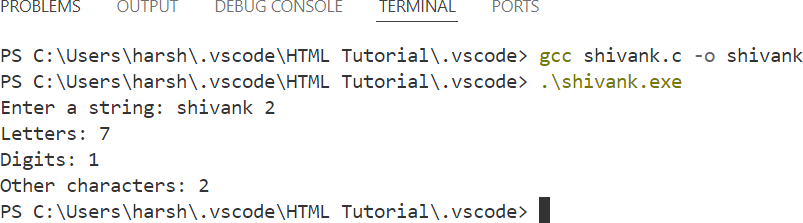
count\_chars\_and\_numbers(input\_string, &letters, &digits, &others); printf("Letters: %d\n", letters);

printf("Digits: %d\n", digits); printf("Other characters: %d\n", others);

return 0;

}

**Output:**

****

**:**

# Practical- 06.

**AIM:** Write a program that uses a function to sort an array of integers.

#include <stdio.h>

int bubble\_sort(int arr[], int n) { for (int i = 0; i < n - 1; i++) {

for (int j = 0; j < n - i - 1; j++) { if (arr[j] > arr[j + 1]) {

// Swap arr[j] and arr[j + 1] int temp = arr[j];

arr[j] = arr[j + 1]; arr[j + 1] = temp;

}

}

}

}

void print\_array(int arr[], int n) { for (int i = 0; i < n; i++) {

printf("%d ", arr[i]);

}

printf("\n");

}

}

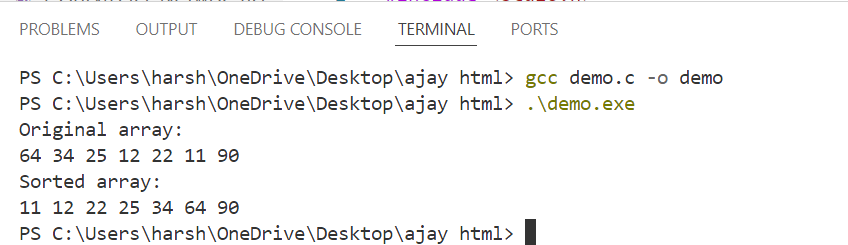
**Output:**

int main() {

int arr[] = {64, 34, 25, 12, 22, 11, 90};

int n = sizeof(arr) / sizeof(arr[0]); printf("Original array: \n"); print\_array(arr, n); bubble\_sort(arr, n); printf("Sorted array: \n"); print\_array(arr, n);

return 0;



# Practical- 07

**AIM:** Write a program to calculate the subject wise and student wise totals and store them as a part of the structure.

#include <stdio.h>

#define MAX\_STUDENTS 100

#define MAX\_SUBJECTS 5 struct Student {

char name[50];

int marks[MAX\_SUBJECTS]; int total;

};

int calculate\_totals(struct Student\* students, int student\_count) { for (int i = 0; i < student\_count; i++) {

students[i].total = 0;

for (int j = 0; j < MAX\_SUBJECTS; j++) { students[i].total += students[i].marks[j];

}

}

return 0;

}

int print\_results(struct Student\* students, int student\_count) { printf("\nResults:\n");

for (int i = 0; i < student\_count; i++) { printf("Student: %s\n", students[i].name); printf("Marks: ");

for (int j = 0; j < MAX\_SUBJECTS; j++) { printf("%d ", students[i].marks[j]);

}

printf("\nTotal: %d\n", students[i].total); printf(" \n");

}

return 0;

}

int main() {

struct Student students[MAX\_STUDENTS]; int student\_count;

printf("Enter the number of students (max %d): ", MAX\_STUDENTS); scanf("%d", &student\_count);

for (int i = 0; i < student\_count; i++) { printf("\nEnter the name of student %d: ", i + 1); scanf("%s", students[i].name);

printf("Enter marks for %d subjects:\n", MAX\_SUBJECTS);

for (int j = 0; j < MAX\_SUBJECTS; j++) { printf("Subject %d: ", j + 1); scanf("%d", &students[i].marks[j]);

}

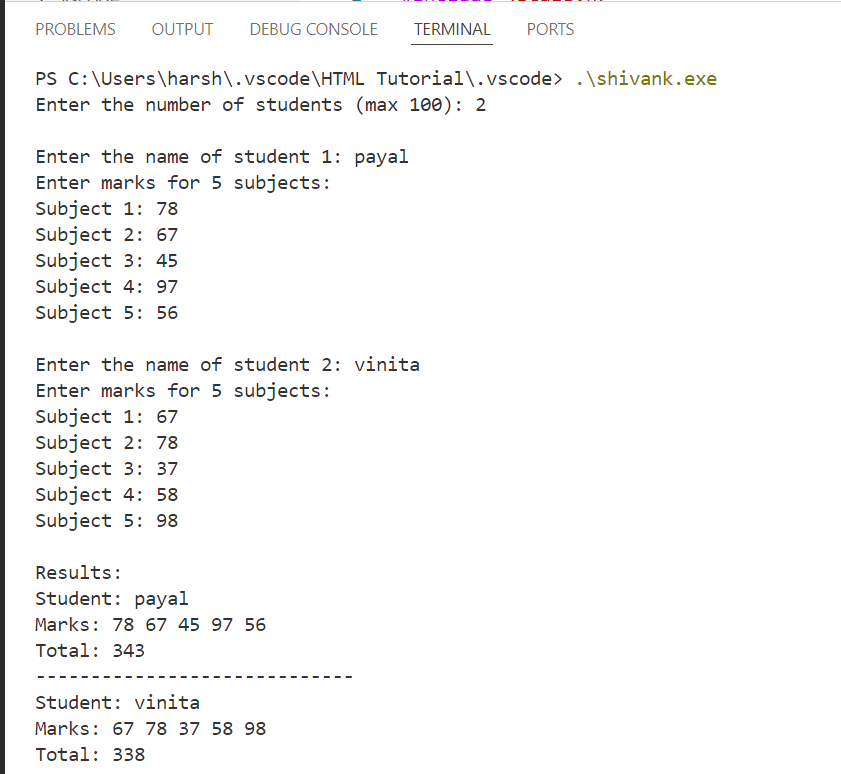
}

calculate\_totals(students, student\_count); print\_results(students, student\_count);

**Output:**

return 0;

}



**AIM:** Write a program to read 10 values to an array variable. Use pointers to locate and display each value.

#include <stdio.h> #define SIZE 10

int main() {

int arr[SIZE]; int \*ptr = arr;

printf("Enter %d values:\n", SIZE); for (int i = 0; i < SIZE; i++) {

printf("Value %d: ", i + 1);

scanf("%d", ptr + i);

}

printf("\nValues in the array:\n"); for (int i = 0; i < SIZE; i++) {

printf("Value %d: %d\n", i + 1, \*(ptr + i));

}

return 0;

}

**Output:**

****

**AIM:** Write a program that uses a table of integers whose size will be specified interactively at runtime.

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

int size;

printf("Enter the size of the array: "); scanf("%d", &size);

int \*arr = (int \*)malloc(size \* sizeof(int)); if (arr == NULL) {

printf("Memory allocation failed!\n"); return 1;

}

printf("Enter %d values:\n", size); for (int i = 0; i < size; i++) {

printf("Value %d: ", i + 1);

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

}

printf("\nValues in the array:\n"); for (int i = 0; i < size; i++) {

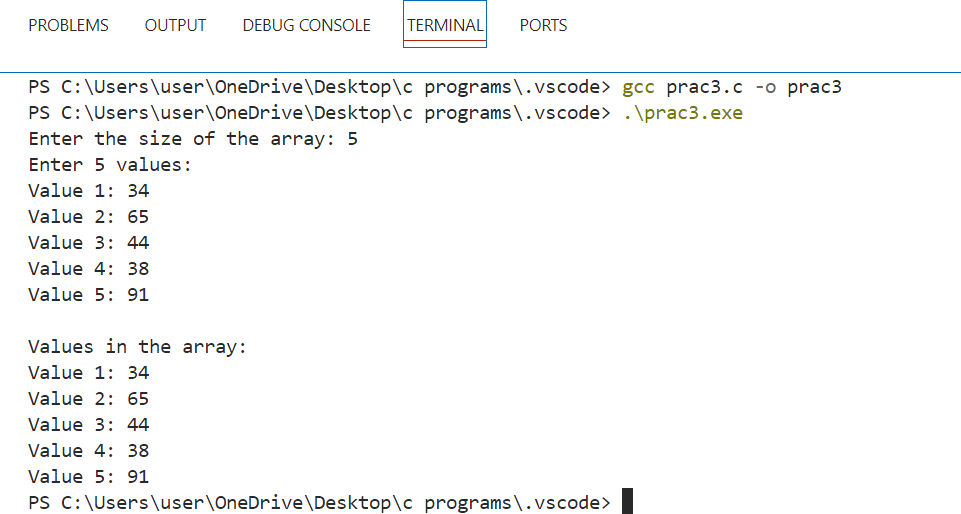
printf("Value %d: %d\n", i + 1, arr[i]);

**Output:**

}

free(arr); return 0;

}



**AIM:** Write a program to store a character string in a block of memory space created by MALLOC and then modify the same to store a larger string.

#include <stdio.h> #include <stdlib.h> #include <string.h>

int main() {

int initial\_size = 20;

char \*str = (char \*)malloc(initial\_size \* sizeof(char)); if (str == NULL) {

printf("Memory allocation failed!\n"); return 1;

}

printf("Enter a string (max %d characters): ", initial\_size 1); fgets(str, initial\_size, stdin);

printf("Original string: %s", str); int new\_size = 100;

str = (char \*)realloc(str, new\_size \* sizeof(char)); if (str == NULL) {

printf("Memory reallocation failed!\n"); return 1;

}

**Output:**

}

printf("Enter a larger string (max %d characters): ", new\_size - 1); fgets(str, new\_size, stdin);

printf("Modified string: %s", str); free(str);

return 0;

