**23CSE102**

**COMPUTATIONAL PROBLEM SOLVING**

**LAB MANUAL**



**Department of computer and communication Engineering**

**Amrita School of Engineering**

**Amrita Vishwa Vidyapeetham, Amaravati Campus**

**Name:**

**Verified By Roll No:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| SNo | Title | Date | Page No | Signature |
| LAB 1 | Flow charts on Sequence | 13-09-24 |  |  |
| a | Flow chart for Difference of two numbers |  | 6 |  |
| b | Flow chart for Swapping of two numbers without using temporary variable |  | 7 |  |
| c | Flow chart for Finding Compound Interest |  | 8 |  |
| d | Flow chart for Converting Lenth in Feet into Centimetres |  | 9 |  |
| e | Flow chart for Finding Average of two numbers |  | 10 |  |
| LAB 2 | Flow charts on Selection statements | 20-09-24 |  |  |
| a | Flow chart for Finding Greatest of 4 numbers |  | 11 |  |
| b | Flow chart for Finding House Electricity Prices |  | 12 |  |
| c | Flow chart for Finding Bus ticket prices |  | 13 |  |
| d | Flow chart for Grading Students |  | 14 |  |
| e | Flow chart for performing Arithmetic operations |  | 15 |  |
| LAB 3 | Flow charts on Looping statements | 11-10-24 |  |  |
| a | Flow chart for Amstrong number or Not |  | 16 |  |
| b | Flow chart for Fibonacci Series |  | 17 |  |
| c | Flow chart for Factorial of a Number |  | 18 |  |
| d | Flow chart for Palindrome number |  | 19 |  |
| e | Flow chart for Reversing a number |  | 20 |  |
| LAB 4 | Simple Python Programs | 18-10-24 |  |  |
| a | Program to add two numbers |  | 21 |  |

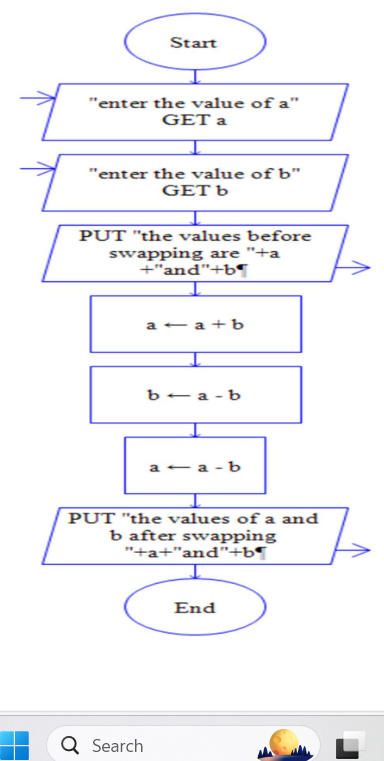
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| b | Program to check number is even or odd |  | 21 |  |
| c | Program to reverse a string |  | 21-22 |  |
| d | Program to find Factorial of a number |  | 22 |  |
| e | Program to find nth Fibonacci Series |  | 23 |  |
| LAB 5 | Programs on Operators | 25-10-24 |  |  |
| a | Program on Arthematic operators |  | 24 |  |
| b | Program on Assissement operators |  | 25 |  |
| c | Program on Comparision operators |  | 26 |  |
| d | Program on Logical operators |  | 27 |  |
| e | Program on Bitwise operators |  | 28 |  |
| f | Program on Special operators |  | 29 |  |
| LAB 6 | Programs on Selection Statements | 15-11-24 |  |  |
| a | Basic BMI Calculator |  | 30 |  |
| b | Grade Evaluation Checker |  |  |  |
| c | Simple Calculator |  |  |  |
| d | Age Group Classification |  |  |  |
| e | Temperature Conversion |  |  |  |

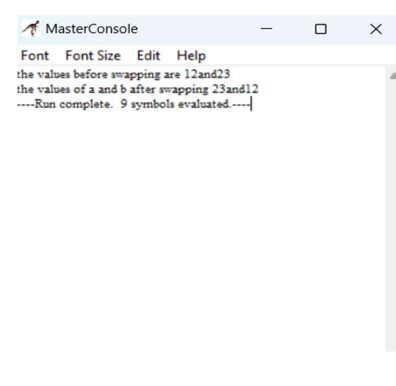
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| f | Leap Year Checker |  |  |  |
| LAB 7 | Programs on Loop + Pattern programs | 22-11-24 |  |  |
| a | Hollow Square Pattern |  |  |  |
| b | Diamond Pattern |  |  |  |
| c | Right Angled Triangle Pattern |  |  |  |
| d | Inverted Right Angled Triangle Pattern |  |  |  |
| e | Numbered Pyramid Pattern |  |  |  |
| f | Hollow Diamond Pattern |  |  |  |
| LAB 8 | Programs on Datatypes | 28-11-24 |  |  |
| a | Program on Numeric Datatype |  |  |  |
| b | Program on Boolean Datatype |  |  |  |
| c | Program on Set Operators |  |  |  |
| d | Dictionary Program |  |  |  |
| e | Program on Lists, Tuples and String |  |  |  |
| LAB 9 | Programs on Arrays | 29-11-24 |  |  |
| a | Program to find the sum and average of array elements |  |  |  |
| b | Program to find largest and smallest elements |  |  |  |
| c | Program to reverse an array |  |  |  |
| d | Program to remove duplicates from array |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| e | Program to search for an element |  |  |  |
| LAB 10 | Programs on Functions and Recursions | 09-12-24 |  |  |
| a | Factorial calculation using recursion |  |  |  |
| b | Fibonacci sequence using recursion |  |  |  |
| c | Sum of digits using recursion |  |  |  |
| d | Power calculation using recursion |  |  |  |
| e | GCD using recursion |  |  |  |

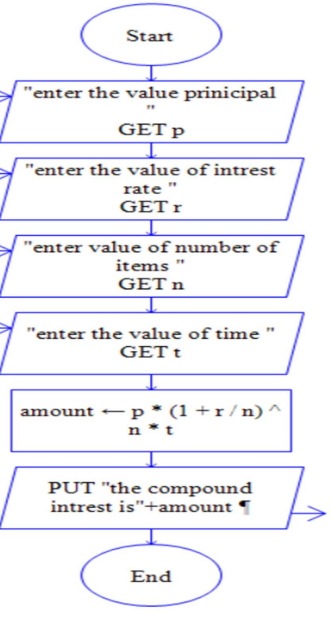
|  |  |
| --- | --- |
| **LAB-1**  1.Flowchart for subtraction of two numbers | 7 |

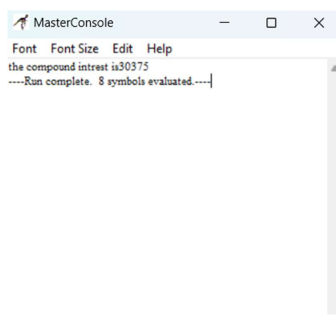
2 . Flowchart for Swapping of two number without using temporary variable





3.Flowchart for finding Compound Interest.





**Observation:**Relational operaters are <,>,=,<=,>= ,==,+! These tell the relation between two numbers by giving output as 0 or 1 ,0 means false and 1 means true .

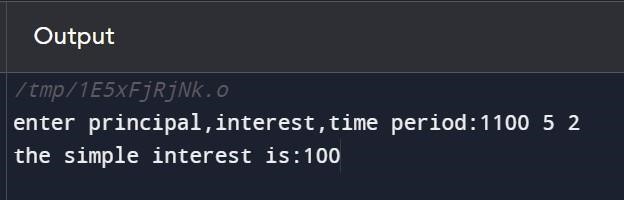
**j) Write a C program to calculate simple interest** #include <stdio.h> int main()

{ int p,i,t;

printf("enter principal ,interest ,time period:"); scanf("%d %d %d",&p,&i,&t);

printf("the simple interest is:%d",p/(1+i\*t)); return 0;

}



**Observation:**first we take the input from the user for the principle ,intrest and time period then Using the printf statement we can get the output of simple intrest(p\*t\*r)/100.

# LAB-2

**a) Write a C program to check weather a number is even or odd using if else statement** #include <stdio.h> int main() { int a;

printf("enter number"); scanf("%d",&a); if(a%2==0){

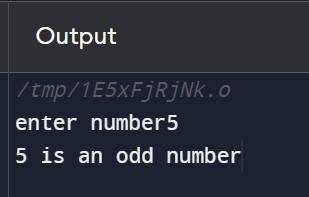
printf("%d is an even number",a);

}else{

printf("%d is an odd number",a);

}

}



**Observation:**to check the even or odd condition we use the condition that a even number is Divisible by 2 and an odd number is not divisible by 2 i.e,if it is even number %2==0 (number When divided by the remainder is zero).

**b) Write a C program to check which number is greatest among three given numbers:**

#include <stdio.h> int main() { int a,b,c;

scanf("%d %d %d",&a,&b,&c);

if(a>b){ if(a>c){

printf("%d is the greatest number",a);

}else{

printf("%d is the greatest number",c);

} }else{ if(b>c){

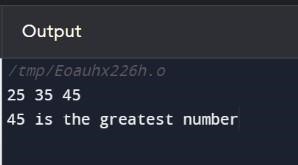
printf("%d is the greatest number",b);

}else{

printf("%d is the greatest number",c);

} } return 0;

}

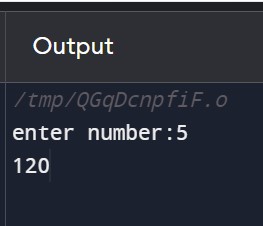


**Observation:**to check the greatest of the three integers we use the nested if else statements In the first if statement we check if a is the greatest number or not by using the <,> operaters , In the else statement we check if b is the greatest number or not.

1. **Write a C program for finding a factorial of a given number:** #include <stdio.h> int main(void) { int factorial=1,n; printf(“enter number:”);

scanf("%d",&n); for(int i=1;i<=n;i++){ factorial\*=i;

}printf("%d",factorial); return 0;}



**Observations:**factorial of a number is the sum of the multiplication of all the integers smaller than

The given number.first we initialize the factorial to zero the use for for loop to update the value

,then print the value of the factorial outside of the loop.

1. **Write a C program for if a number is Armstrong or not:**

#include <stdio.h> int main() { int n,a,b,sum=0; scanf("%d",&n); int ori\_num=n; while(n>0){ a=n/10; b=n%10; sum=sum+(b\*b\*b); n=a; }

|  |  |
| --- | --- |
| if(ori\_num==sum){  printf("%d is an armstrong number",ori\_num);  }else{  printf("%d is not an armstrong number",ori\_num);  } return 0;  }      **Observation:**three digit armstrong number is a number that is equal to the sum of cubes of its Own digits.then while using while loop we update the sum to sum plus the cude of each digit.  Outside the loop we check if the sum is equal the original or not using the if statement ,if true It is a armstrong number else it is not an armstrong number.  **e) Write a C program to print the day of the week using switch case** #include <stdio.h> int main() { int week;  printf("Enter week number(1-7): "); scanf("%d", &week); switch(week)  { case 1: printf("Monday"); break; case 2:  printf("Tuesday"); | 18 |

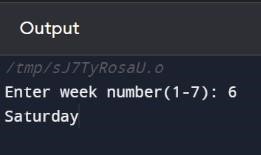
break; case 3: printf("Wednesday"); break; case 4: printf("Thursday"); break; case 5: printf("Friday"); break;

case 6: printf("Saturday"); break;

case 7: printf("Sunday"); break; default:

printf("Please enter week number between 1-7.");

} return 0;}



**Observation:**using the switch case we ouput the day of the week ,when the number of the day is Inputed the day is given as output ,i.e, when 1 is given as input in 1st case Monday is given as Output.

**f) Write a C program for checking if a number is palindrome or not**

#include <stdio.h>

int main(){ int a,b,n,reverse=0; scanf("%d",&n); int ori\_num=n; while(n>0){ a=n/10; b=n%10; n=a;

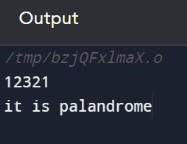
reverse=reverse\*10+b; }if(ori\_num==reverse){ printf("it is palindrome");

}else{

printf("it not palindrome");

} return 0;

}



**Observation:**palindrome means a number which remains the same even when reversed ,so to check we use the while loop (n>0),first we initialize reverse to 0,then update it as reverse\*10+the last digit (n%10),then update the n=n/10.outside the loop we check if the number is equal yo the reverse or not using the of statement and print the repective output.

**g) Write a C program to generate all the prime numbers between 1 and n**

#include <stdio.h>

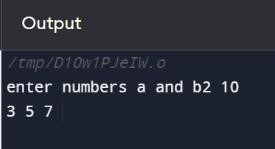
int main(void) { int a,b;

printf("enter numbers a and b"); scanf("%d %d",&a,&b); for(int i=a;i<=b;i++){ for(int j=2;j<=(i/2)+1;j++){ if(i%j==0){ break;

} else if(j==(i/2)+1){ printf("%d ",i); }

} } return 0;

}



**Observation:**to print the prime numbers from 1 to n we use two for loops ,one for transvering throngh 1 to n numbers and the other for transversing from 2 to i/2 ,in the loop we we use if statement to check the divisibility of the number ,if it is not divisible by any number we print the number .

1. **Write a C program to print the multiplication table of a given number n up to a given value** #include <stdio.h> int main()

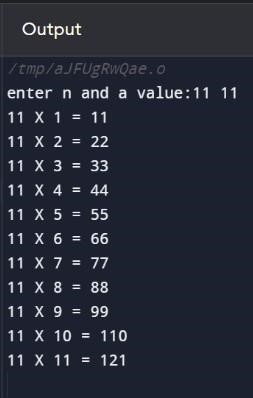
{

int n,a;

printf("enter n and a value:"); scanf("%d %d",&n,&a); for(int i=1;i<=a;i++){

printf("%d X %d = %d\n",n,i,n\*i); } return 0;

}



**Observation:**first we take the input from the user of the table to printed (n)and the number till the table should be printed(a),then using the for loop we print the table.

1. **Write a C program to count number of digits in given integer**

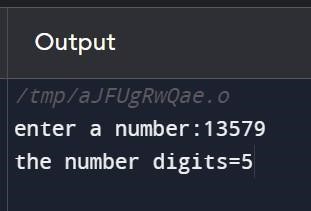
#include <stdio.h>

int main() { int n,a,count=0; printf("enter a number:"); scanf("%d",&n); while(n>0){

a=n/10; n=a; count=count+1;

}printf("the number digits=%d",count); return 0;

}



**Observation:**first we should initialize count variable to 0 and using the while loop (n>0) update the count +1 so the number of times the loop runs is the number of digits the number has.

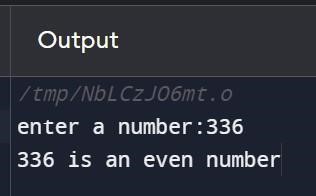
1. **Write a C program to check odd or even using conditional operator** #include <stdio.h>

int main() { int n;

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

n%2==0?printf("%d is an even number",n):printf("%d is an odd number",n); return 0;

}



**Observation:** to check the even or odd condition we use the condition that a even number is

Divisible by 2 and an odd number is not divisible by 2 i.e,if it is even number %2==0 (number

When divided by the remainder is zero).

1. **Write a C Program Fibonacci Series up to n number of terms**

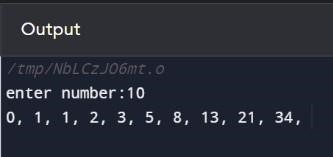
#include <stdio.h> int main(){ int n;

printf("enter number:"); scanf("%d",&n); int t1=0,t2=1; int next\_term=t1+t2; printf("%d, %d, ",t1,t2); for(int i=3;i<=n;++i){

printf("%d, ",next\_term); t1=t2; t2=next\_term; next\_term=t1+t2;

} return 0;

}



|  |  |
| --- | --- |
|  | 25  **Observation:** fibonacci is a sequence in which each number is the sum of the two preceding ones,its starts from 0,1,1,2….we initialize t1=0 and t2=1 then print them ,using the for loop we update the next term to t1 +t2 and print the next term . |
| **l)** | **Write a C Program to check given year is leap year or not**.  #include <stdio.h> int main() { int a;  printf("enter year:"); scanf("%d",&a); if(a%400==0){  printf("it is a leap year");  }else if (a%100==0){ printf("it is not a leap year");  }else if (a%4==0){ printf("it is a leap year");  }else{ printf("it is not a leap year");  } return 0;}    **Observation:**a year is leap year if it is divisible by 4,100,400.so we use the if else if statements to determine if it is a leap year or not. |

# LAB 3

**a)**  **Write a C program to read two integer numbers and find their sum, difference and product using separate functions**

#include <stdio.h> int sum(int a,int b){ return a+b;

}

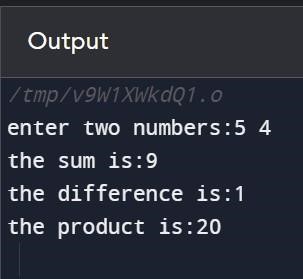
int difference(int a,int b){ return a-b;

} int product(int a,int b){ return a\*b;

} int main() { int a,b;

printf("enter two numbers:"); scanf("%d %d",&a,&b); printf("the sum is:%d\n",sum(a,b)); printf("the difference is:%d\n",difference(a,b)); printf("the product is:%d\n ",product(a,b));

return 0;}



**Observation:**we use three different functions to get the sum, difference and product.the first function sum retuns the value a+b and difference function returns a-b and product function returns a\*b.the in the main function these functions are called to return the value.

**b) Design a function “Cube” which returns the cube of any given number**

#include <stdio.h> int cube(int a){ return a\*a\*a;

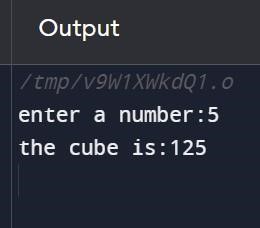
} int main() { int a;

printf("enter a number:"); scanf("%d",&a);

printf("the cube is:%d\n",cube(a));

return 0;

}



**Observation:**we use the a function named cube to retun the value a\*a\*a by passing the integer a to the function .by calling the function in the main function we can get the output.

**c) Design two functions “area” & “perimeter” which will return area and perimeter of a rectangle.**

#include <stdio.h> int area(int a,int b){ return a\*b;

}

int perimeter(int a,int b){ return 2\*(a+b);

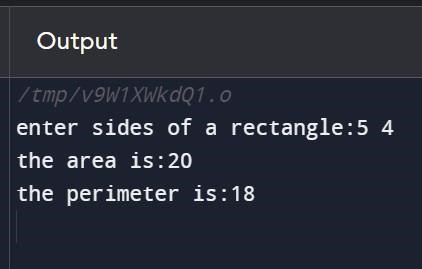
} int main() { int a,b;

printf("enter sides of a rectangle:"); scanf("%d %d",&a,&b);

printf("the area is:%d\n",area(a,b)); printf("the perimeter is:%d\n",perimeter(a,b));

return 0;

}



**Observation:**we declare two functions namely area and perimeter ,area function will return l\*b by passing the length and breath to the function .perimeter function will return 2\*(l+b) by passing the length and breath to the function.

**d) Write a C program for inbuilt functions of Math.h header file**

#include <stdio.h>

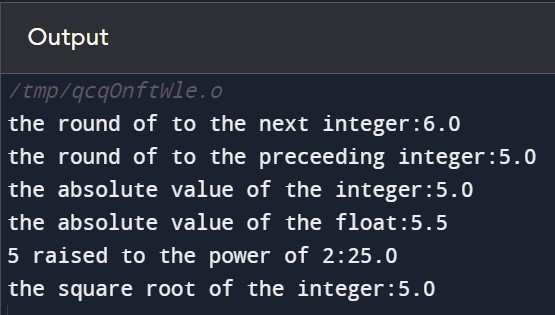
#include<math.h>

#include<stdlib.h>

int main(){

printf("the round of to the next integer:%.1f\n",ceil(5.5)); printf("the round of to the preceeding integer:%.1f\n",floor(5.5)); printf("the absolute value of the integer:%.1f\n",abs(-5.5)); printf("the absolute value of the float:%.1f\n",fabs(-5.5)); printf("%d raised to the power of %d:%.1f\n",5,2,pow(5,2)); printf("the square root of the integer:%.1f\n",sqrt(25)); return 0;

}



**Observation:**math.h is a header file in which pre-defined math functions are inbuilt like ceil,floor,abs,fabs,pow,sqrt and many.ceil and floor will round off the given number .abs and fabs will return the absolute value of the given number .pow function will give the a raised to the power of b.sqrt will give the square root of a number.

**e) Write a C program for inbuilt functions of Ctype.h header file**

#include <stdio.h>

#include <ctype.h>

int main() { int ch='%'; printf("%d\n",isalnum(ch)); printf("%d\n",isalpha('a')); printf("%d\n",islower('A')); printf("%d\n",isupper('D'));

return 0;

}



**Observation:**ctype.h is a header file in which pre-defined functions which are used to classify and transform the characters are present like isalnum,isalpha,islower,isupper.

**f) Write a C program to Find Sum of Natural Numbers Using Recursion**

#include <stdio.h> int sum(int n){ if(n!=0){ return n+sum(n-1);

}

else{ return n;

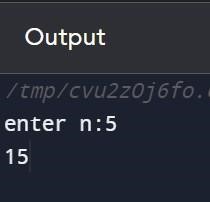
}

}

int main() { int n; printf("enter n:"); scanf("%d",&n); printf("%d",sum(n));

return 0;

}



**Observation:**a recurssion process is a process in which the program repeats a certain section of code i.e, a function will calls itself .we declare a sum functions which will repeatedly call itself till a counter condtion is encountered i.e, when n=0.

**g) Write a C program to Reverse a sentence using recursion**

#include <stdio.h> void reverse(){

char c; scanf("%c",&c); if(c!='\n'){ reverse(); printf("%c",c);

}

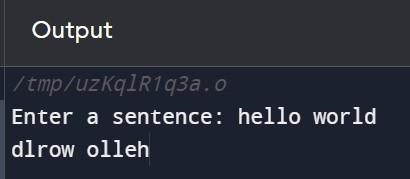
}

int main() {

printf("Enter a sentence: "); reverse();

return 0;

}



**Observation:**a reverse function is declared which calls itself it it encounter \n and then it prints the sentence in the reverse order.

**h) Write a C Function to calculate the factorial of a number**

#include <stdio.h> int fact(int n){ if(n!=0){

return n\*fact(n-1);

} else{ return 1;

}

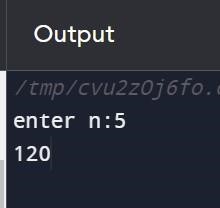
}

int main() { int n;

printf("enter n:"); scanf("%d",&n); printf("%d",fact(n));

return 0;

}



**Observation:**factorial of a number is the multiplication of all the integers smaller than the given number .we declare the factorial function which calls itself till it encounters n=0.and multipies n\*factorial of n-1.

**i) Write a C Function to swap to integers**

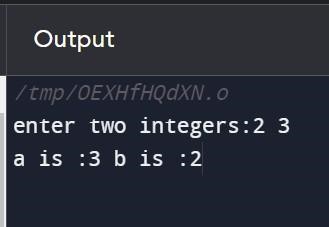
#include <stdio.h> int swap(int a,int b){ int c=a; a=b; b=c; printf("a is :%d ",a); printf("b is :%d",b); return 0;

}

int main() { int a,b; printf("enter two integers:"); scanf("%d %d",&a,&b); swap(a,b);

return 0;

}



**Observation:**we use a swap function in which we use the temporary variable to swap the integers.

First we store the value of a in the temporary variable then the value of b is saved in a ,in b the a value which is saved in temporary variable is stored.

**j) Write a program in C to convert a decimal number to binary using recursion**

include <stdio.h> int binary(int decimal){ if(decimal==0){ return 0;

}else{ return(decimal%2+10\*binary(decimal/2)); } } int main(){ int decimal;

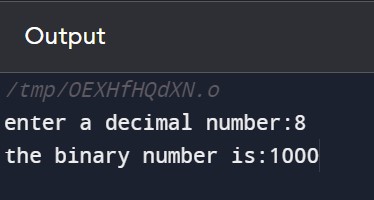
printf("enter a decimal number:");

scanf("%d",&decimal);

printf("the binary number is:%d",binary(decimal));

return 0;

}



**Observation:**a function binary is declared in which a decimal number is passed it returns a binary number by (decimal%2+10\*binary(decimal/2)) by recursion.

# LAB-4

**a) Write a Program to take n values from the user and store them in an array and print the elements stored in the array**

#include <stdio.h>

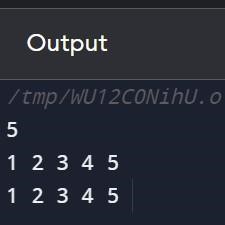
int main() { int n;

scanf("%d",&n); int arr[n]; for(int i=1;i<=n;i++){ scanf("%d",&arr[i]); printf("%d ",arr[i]);

}

return 0;

}



|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values, printing each value as it is entered.

**b) Write a Program Read n number of values in an array and display 5th element in the array** #include <stdio.h>

int main() { int n; scanf("%d",&n); int arr[n]; for(int i=1;i<=n;i++){ scanf("%d",&arr[i]);

}

printf("%d ",arr[5]);

return 0;

|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values,then the 5th element which is in the 4th index is printed.

}



**c) Write a Program Read n number of values in an array and display it in reverse order** #include <stdio.h>

int main() { int num; printf("enter a number:"); scanf("%d",&num); int arr[num]; for(int i=1;i<=num;i++){ scanf("%d",&arr[i]);

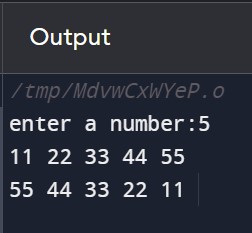
}

for(int i=num;i>=1;i--){ printf("%d ",arr[i]);

}

return 0;

}



|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values, then a separate for loop is used to print the array in the reverse order by starting the loop from n to 1.

**d) Write a program prints the sum of elements of an array.** #include <stdio.h>

int main() {

int n,sum=0;

scanf("%d",&n); int arr[n]; for(int i=1;i<=n;i++){ scanf("%d",&arr[i]);

sum=sum+arr[i];

}

printf("%d ",sum);

return 0;

}



|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values,and sum is equal to sum +arr[i] then the sum is printed outside the loop.

**e) Write a program finds the highest and lowest elements in an array.** #include <stdio.h>

int main() { int n,high,low;

printf("enter the size of the array:");

scanf("%d",&n);

int arr[n]; for(int i=0;i<n;i++){ scanf("%d",&arr[i]);

} low=arr[0]; high=arr[0]; for(int i=1;i<n;i++){ if(arr[i]>high){

high=arr[i];

}else if(arr[i]<low){ low=arr[i];

}

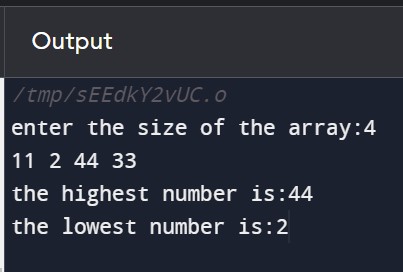
}

printf("the highest number is:%d\n",high);

printf("the lowest number is:%d",low);

return 0;

}



|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values.we initialize low and high to 0th index value .then while using the for loop and the if else statements check if the two integers are greater or not if arr[1] is greater than the high store the element in the high variable .outside the loop print the highest and lowest values that are stored in the low and high variables.

**f) Write a Program to find second largest element in a one-dimensional array** #include <stdio.h>

int main() { int n,high,high2; printf("enter the size of thr array:");

scanf("%d",&n);

int arr[n]; for(int i=0;i<n;i++){

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

}

high2=high=arr[0]; for(int i=1;i<n;i++){

if(arr[i]>high){ high2=high; high=arr[i];

}else if(arr[i]>high2 && arr[i]<high){ high2=arr[i];

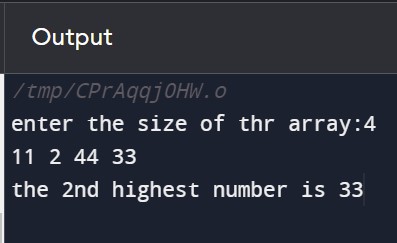
}

}

printf("the 2nd highest number is %d",high2);

return 0;

}



**Observation:** The code begins by prompting the user to input the size of the array and then proceeds to initialize an array of that size. Using a loop, the code iterates through the array to find the second-highest number. It maintains variables **high** and **high2** to track the highest and second-highest values.

**g) Write a Program Read n number of values in an array and modify 3rd element in the array and display array.** #include <stdio.h>

int main() { int n,a;

printf("enter the size of the array:");

scanf("%d",&n);

int arr[n]; for(int i=0;i<n;i++){ scanf("%d",&arr[i]);

} printf("enter a modifed 3rd element:");

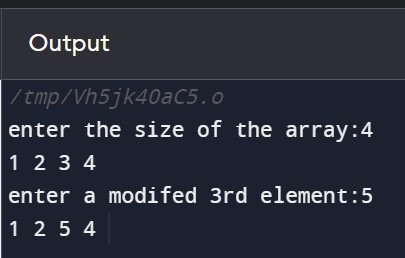
scanf("%d",&a); arr[2]=a;

for(int i=0;i<n;i++){ printf("%d ",arr[i]);

}

return 0;

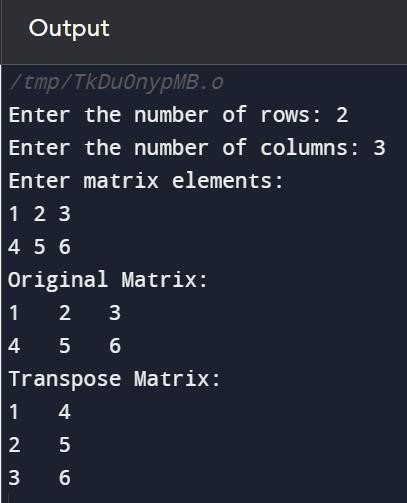
}



|  |
| --- |
| **arr** |

**Observation:** The code prompts the user to input an integer **n** and then creates an array of size **n**. It uses a loop to populate the array with user-inputted values.the ask the user to input the modified 3rd element then update the value and print the whole array again .

|  |  |
| --- | --- |
| **LAB 5**        **a) Write a C Program to Print Transpose of a Matrix** #include <stdio.h>    int main() { int a, cols;    // Input matrix dimensions printf("Enter the number of rows: "); scanf("%d", &a);  printf("Enter the number of columns: "); scanf("%d", &cols);  int matrix[a][cols];    // Input matrix elements  printf("Enter matrix elements:\n");  for (int i = 0; i < a; i++) for (int j = 0; j < cols; j++) scanf("%d", &matrix[i][j]);    // Print original matrix printf("Original Matrix:\n"); for (int i = 0; i < a; i++) { for (int j = 0; j < cols; j++) printf("%d\t", matrix[i][j]);  printf("\n");  }    // Print transpose of matrix printf("Transpose Matrix:\n"); for (int i = 0; i < cols; i++) { for (int j = 0; j < a; j++) printf("%d\t", matrix[j][i]);  printf("\n");  }    return 0;  } | 41 |



**Observation:** first we take rows and columns as the input and declare a matrix and take the elements of the matrix as input from the user. Then we print the original matrix end the transpose matrix by reversing the rows and columns.

**b) Write a C Program Add Two Matrix Using Multi-Dimensional Arrays** #include <stdio.h>

void add(int a,int b,int arr1[a][b],int arr2[a][b]){ int sum[a][b]; for(int i=0;i<a;i++){

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

sum[i][j]=arr1[i][j]+arr2[i][j];

}

}

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

for(int j=0;j<b;j++){ printf("%d ",sum[i][j]);

}

printf("\n");

}

}

int main(){

int a,b;

printf("enter the size of the array(row-column):");

scanf("%d %d",&a,&b);

int arr1[a][b];

printf("enter the elementsof the 1st matrix:\n"); for(int i=0;i<a;i++){

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

scanf("%d",&arr1[i][j]);

}

printf("\n");

}

int arr2[a][b];

printf("enter the elements of the 2nd matrix:\n"); for(int i=0;i<a;i++){

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

scanf("%d",&arr2[i][j]);

}

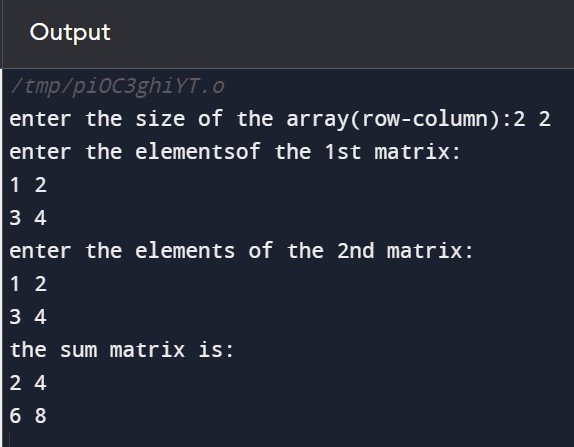
printf("\n");

}

printf("the sum matrix is:\n"); add(a,b,arr1,arr2);

return 0;

}



**Observation:** we take the size of both the matrices as A and B and take the input of the elements in A&B we define a function add. The add function stores the values of the sum of the elements in A and B.

**c) Write a C Program Multiply to Matrix Using Functions**

|  |  |
| --- | --- |
| #include <stdio.h>  void multiply(int row1,int col1,int row2,int col2,int a[row1][col1],int b[row2][col2]){ int mul[row1][col2]; for(int i=0;i<row1;i++){ for(int j=0;j<col2;j++){ int sum=0;  for(int k=0;k<col1;k++){ sum+=a[i][k]\*b[k][j];    } mul[i][j]=sum;  }  }  printf("the result of the multiplication of the matrices:\n"); for(int i=0;i<row1;i++){ for(int j=0;j<col2;j++){ printf("%d ",mul[i][j]);  }  printf("\n");  }  }    int main(){ int row1,col1,row2,col2;  printf("enter the size of the 1st matrix(row-column):\n"); scanf("%d %d",&row1,&col1);  printf("enter the size of the 2nd matrix(row-column):\n");  scanf("%d %d",&row2,&col2);  //check if multiplication is possible  if(col1!=row2){ printf("multiplication is not possible.\n"); return 0;  }  int a[row1][col1],b[row2][col2];  printf("enter the elements of the 1st matrix:\n"); for(int i=0;i<row1;i++){ for(int j=0;j<col1;j++){ scanf("%d",&a[i][j]);  }  printf("\n");  }  printf("enter the elements of the 2nd matrix:\n"); | 44 |

for(int i=0;i<row2;i++){ for(int j=0;j<col2;j++){ scanf("%d",&b[i][j]);

}

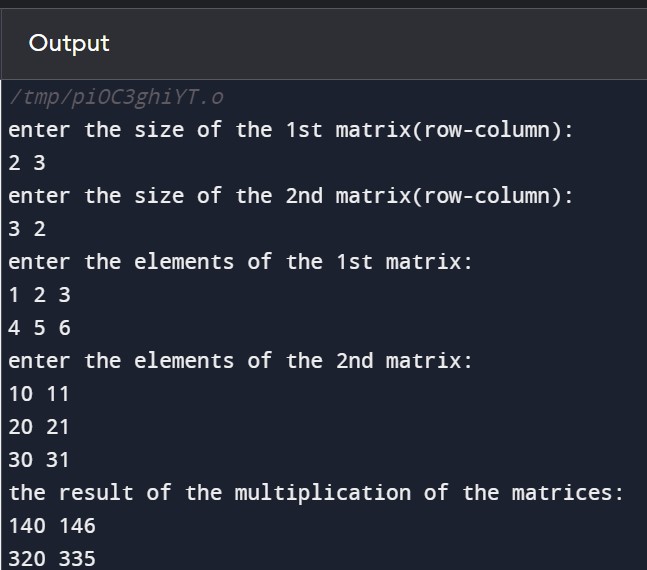
printf("\n");

}

multiply(row1,col1,row2,col2,a,b);

return 0;

}



**Observation:** we take the input are the first matrix as row one and column one, and the size of the second matrix as row two and column two. Then we check if multiplication is possible for the two matrices by comparing the column one and row 2. If multiplication is possible we take the input for both the matrices an input them two multiply function. This function stores it's sentence matrix in multiply matrix imprint the final matrix as output.

1. **Write a C Program to declare two 3D arrays and print the two matrixes** #include <stdio.h>

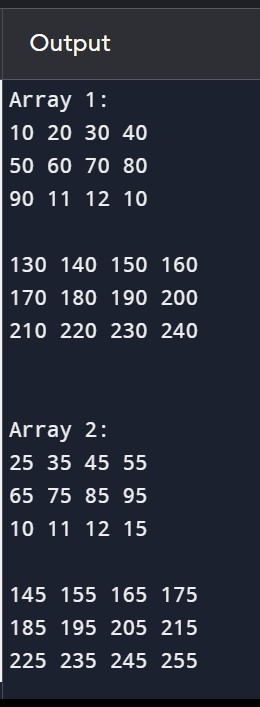
int main() {

// Declare and initialize two 3D arrays with different values int array1[2][3][4] = {

{{10, 20, 30, 40}, {50, 60, 70, 80}, {90, 100, 110, 120}},

{{130, 140, 150, 160}, {170, 180, 190, 200}, {210, 220, 230, 240}} };

|  |  |
| --- | --- |
| int array2[2][3][4] = {  {{25, 35, 45, 55}, {65, 75, 85, 95}, {105, 115, 125, 135}},  {{145, 155, 165, 175}, {185, 195, 205, 215}, {225, 235, 245, 255}}  };  // Print the elements of the first 3D array printf("Array 1:\n"); for (int i = 0; i < 2; i++) { for (int j = 0; j < 3; j++) { for (int k = 0; k < 4; k++) { printf("%d ", array1[i][j][k]);  }  printf("\n");  }  printf("\n");  }    printf("\n");    // Print the elements of the second 3D array printf("Array 2:\n"); for (int i = 0; i < 2; i++) { for (int j = 0; j < 3; j++) { for (int k = 0; k < 4; k++) { printf("%d ", array2[i][j][k]);  }  printf("\n");  }  printf("\n");  }    return 0;  } | 46 |



**Observation:** we declare 2 matrices namely A and B with certain values. Then we print these two matrices using for loops. The first for loop is for the number of matrices , the second for loop is for the number of rows and the third for loop it's for the number of columns.

1. **Write a C Program to find sum of all elements of each row of a matrix** #include <stdio.h>

int main() { int rows, cols;

// Input matrix dimensions printf("Enter the number of rows: "); scanf("%d", &rows);

printf("Enter the number of columns: "); scanf("%d", &cols);

int matrix[rows][cols];

// Input matrix elements printf("Enter matrix elements:\n"); for (int i = 0; i < rows; i++) for (int j = 0; j < cols; j++) scanf("%d", &matrix[i][j]);

// Calculate and print the sum of each row printf("Sum of each row:\n");

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

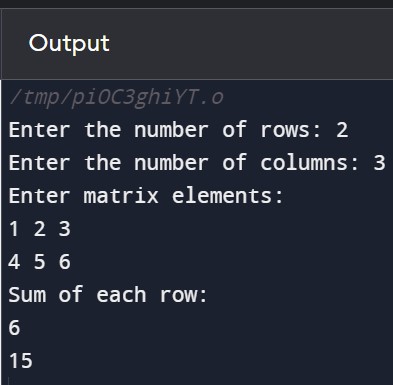
for (int j = 0; j < cols; j++){ sum += matrix[i][j];}

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

}

return 0;

}



**Observation:** to print the sum of each row first we take the input of the elements of the matrix then we use for loop to add the elements in each row and print them in order.

**f) Write a C Program to read a matrix and find sum, product of all elements of two-dimensional**

**(matrix) array**

#include <stdio.h>

int main() { int rows, cols;

// Input matrix dimensions printf("Enter the number of rows: "); scanf("%d", &rows);

printf("Enter the number of columns: "); scanf("%d", &cols);

int matrix[rows][cols];

// Input matrix elements printf("Enter matrix elements:\n"); for (int i = 0; i < rows; i++)

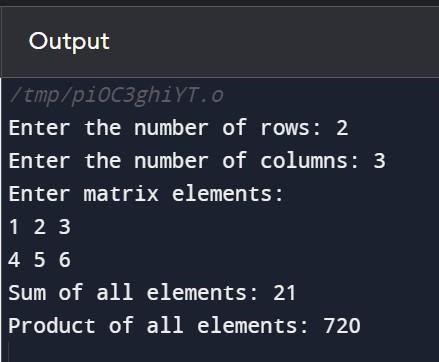
for (int j = 0; j < cols; j++) scanf("%d", &matrix[i][j]);

// Calculate the sum and product of all elements int sum = 0, product = 1; for (int i = 0; i < rows; i++) for (int j = 0; j < cols; j++) { sum += matrix[i][j]; product \*= matrix[i][j]; }

// Print the result

printf("Sum of all elements: %d\n", sum); printf("Product of all elements: %d\n", product);

return 0;}



**Observation:** after inputting the elements in the matrix we calculate the sum and product of all the elements in the matrix using a for loop and storing the sum of digits in a variable sum and the product of all the elements in a variable called product. And finally printing the sum and product as output.

**g) Write a C program to read a matrix and print its diagonals** #include <stdio.h>

int main() { int rows, cols;

// Input matrix dimensions printf("Enter the number of rows: "); scanf("%d", &rows);

printf("Enter the number of columns: ");

scanf("%d", &cols); int matrix[rows][cols];

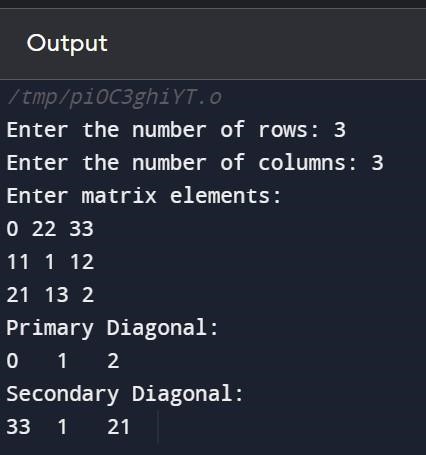
// Input matrix elements printf("Enter matrix elements:\n"); for (int i = 0; i < rows; i++) for (int j = 0; j < cols; j++) scanf("%d", &matrix[i][j]);

// Print the diagonals printf("Primary Diagonal:\n"); for (int i = 0; i < rows; i++) printf("%d\t", matrix[i][i]);

printf("\nSecondary Diagonal:\n"); for (int i = 0; i < rows; i++) printf("%d\t", matrix[i][rows - 1 - i]);

return 0;

}



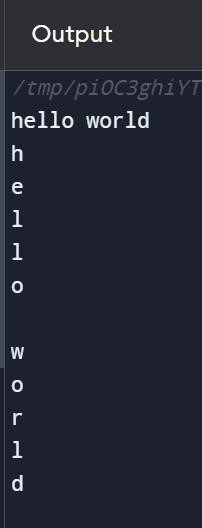
**Observation:** after taking the elements in the matrix as the input .first we print the primary diagonal using a for loop keeping the rows and columns constant, next we print the secondary diagonal keeping the rows as I and columns as row -1 - i in a for loop.

# LAB 6

1. **Declare string and print character by character using loop** #include <stdio.h> int main(){ char str[100]; fgets(str,100,stdin); for(int i=0;str[i]!='\0';i++){ printf("%c\n",str[i]);

} return 0;

}



**Observation:** we input the string using fgets function I'm using for loop we print each character in a new line.

1. **Program to create, read and print an array of strings in C** #include <stdio.h>

int main() { int n; printf("the no of string:"); scanf("%d",&n); char strings[n][50];

getchar();

// Create and read strings into the array printf("Enter %d strings:\n", n); for (int i = 0; i <n; i++) { printf("Enter string %d: ", i + 1); fgets(strings[i], 50, stdin);

// Remove the newline character from fgets if present

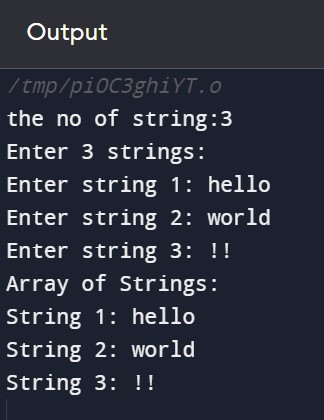
if (strings[i][strlen(strings[i]) - 1] == '\n') { strings[i][strlen(strings[i]) - 1] = '\0'; }

}

// Print the array of strings printf("\nArray of Strings:\n"); for (int i = 0; i < n; i++) { printf("String %d: %s\n", i + 1, strings[i]); }

return 0;

}



**Observation:** be declared the number of strings we wanna input and the size of each string using getchar function we remove the new line character then we create and read strings into the array .

Then we remove the new line character from fgets if present. And then finally print the array.

**c) Program to find occurrence of a character in an input string in C** #include <stdio.h>

void checkchar(char str[],char n){ for(int i=0;str[i]!='\0';i++){ if(str[i]==n){ printf("charecter is present!!"); return;

}

}

printf("charecter not found:(");

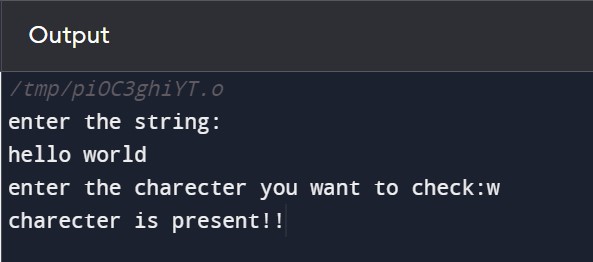
} int main(){ char str[100],n;

printf("enter the string:\n"); fgets(str,100,stdin);

printf("enter the charecter you want to check:");

scanf("%c",&n); checkchar(str,n); return 0;

}



**Observation:** To find the occurrence of a character in array string we used the function named chick car and pass the string into the function. With transfers through all the characters in this string checking each character with the character we want if the character is found we return character found else character not found.

**d) Program to get the indexes of a particular characters in C** #include <stdio.h>

void checkchar(char str[],char n){ int a=0;

for(int i=0;str[i]!='\0';i++){ if(str[i]==n){ printf("charecter is present at %d\n",i+1);

a=1;

}

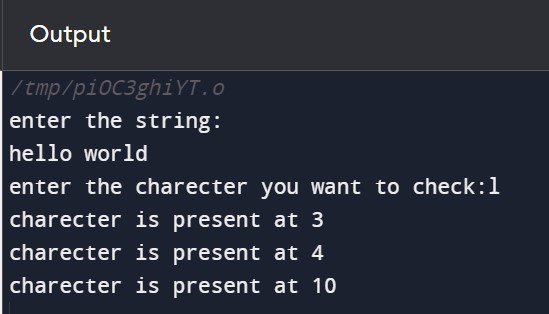
} if(a==0){ printf("element not found:(");

} } int main(){ char str[100],n;

printf("enter the string:\n"); fgets(str,100,stdin);

printf("enter the charecter you want to check:"); scanf("%c",&n); checkchar(str,n); return 0;

}



**Observation: :** To find the occurrence of a character in array string we used the function named chick car and pass the string into the function. With transfers through all the characters in this string checking each character with the character we want if the character is found we return character found with the index it is present in else character not found.

**e) Program to eliminate all vowels from the string in C** #include <stdio.h> int main(){

char str[100]; printf("enter the string:"); fgets(str,100,stdin);

char newstr[100] = "";

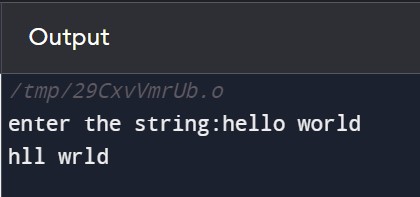
int j=0; for(int i=0;str[i]!='\0';i++){

if(str[i]=='a' || str[i]=='e' || str[i]=='i' || str[i]=='o' || str[i]=='u' ){

}else{ newstr[j]=str[i]; j++;

} } newstr[j]='\0'; puts(newstr); return 0;

}



**Observation:** after inputting the string free transfers through every character in the string and check if it is a vowel, if yes we don't print it else we print to the character.

**f) C program to check a string is palindrome or not**

#include <stdio.h> #include <string.h> void palindrome(char str[]){

int lenght=strlen(str);

int first=0; int last=lenght-1;

int flag=0; while(first<=last){

if(str[first]==str[last]){

first++;

last--;

}else{ flag=1; break;

} } if(flag!=1){

printf("it is palindrome");

}else{ printf("it is not palindrome"); }

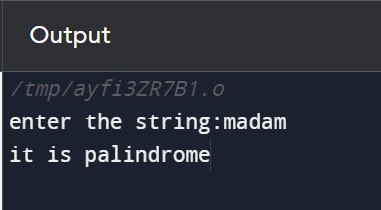
}

int main(){

char str[100]; printf("enter the string:"); fgets(str,100,stdin); if (str[strlen(str) - 1] == '\n') { str[strlen(str) - 1] = '\0';

} palindrome(str); return 0;

}



**Observation:** could check if a string is palindrome or not first we find the length of the string using the strlen function. Then initialize the variables first last and flag. Using the while loop we check is the first character and the last character are same , then we update the values of the first and last to the second last and the second element and check if they're same or not we repeat the process till we reach the middle character. If if the flag it's one we print it is not palindrome else it is palindrome.

**g) C program to print all VOWEL characters separately in given string.**

#include <stdio.h>

#include <string.h>

void printVowels(char str[]) { printf("Vowel characters in the string: "); for (int i = 0; i < strlen(str); ++i) { char ch = str[i];

// Check if the character is a vowel (case-insensitive) if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' || ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') { printf("%c ", ch);

}

}

printf("\n");

}

int main() { char str[100]; printf("Enter the string: "); fgets(str, sizeof(str), stdin);

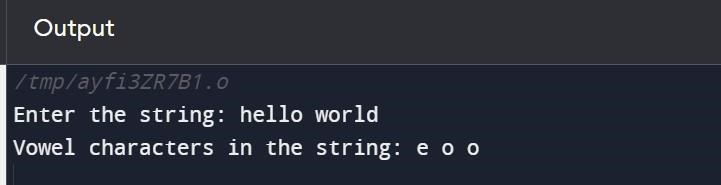
// Remove the newline character if it exists

if (str[strlen(str) - 1] == '\n') { str[strlen(str) - 1] = '\0';

}

printVowels(str); return 0;

}



**Observation:** after checking the character is a vowel or not B print the vowels in the string in order.

# LAB 7

**a) C program to copy a string to another string using recursion** #include <stdio.h>

void copystring(char main[],char copied[],int i){ if(main[i]=='\0'){ copied[i]='\0'; return 0;

}

copied[i]=main[i];

return copystring(main,copied,++i);

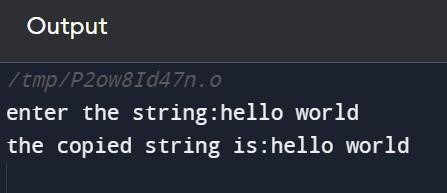
} int main(){ char main[100],copied[100]; printf("enter the string:"); fgets(main,100,stdin);

const int i=0;

copystring(main,copied,i);

printf("the copied string is:%s",copied); return 0;

}



**Observation:**the copystring function is a recursive function that takes the mainstream and the copied string and the index as parameters it copies one character at a time and recursively calls itself with the next index until it reaches the null Terminator.

**b) C program to find the GCD (Greatest Common Divisor) of given numbers using recursion** #include <stdio.h> int GCD(int a,int b){ if(b==0){ return a;

}

return GCD(b,a%b);

}

int findGCDofarray(int arr[],int n){ if(n==1){ return arr[0];

}

return GCD(arr[n-1],findGCDofarray(arr,n-1));

} int main(){ int n; printf("enter n:");

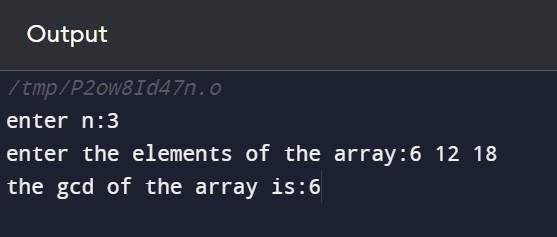
scanf("%d",&n); int arr[n];

printf("enter the elements of the array:"); for(int i=0;i<n;i++){ scanf("%d",&arr[i]);

}

printf("the gcd of the array is:%d",findGCDofarray(arr,n)); return 0;

}



**Observation:** this program first takes the number of elements in the array and then takes input of the each element in the array. It then uses a recursive function to find the greatest common divisor of all the numbers in the array. The findgcdofarray function clear cursive function that aims to find the greatest common divisor for an array of numbers and the base case I spent there is one element in the array.

**c) C program to read a value and print its corresponding percentage from 1% to 100% using recursion**

#include <stdio.h> float percentage(int n,float i){ if(i>100){ return;

}

float result=(i/100)\*n;

printf("%.1f percentage of %d is %.1f\n",i,n,result); return percentage(n,++i);

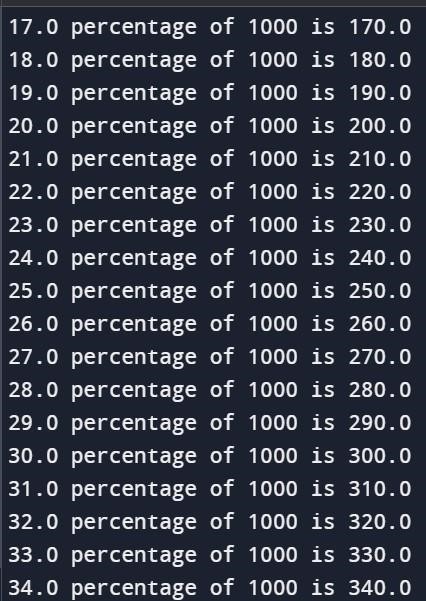
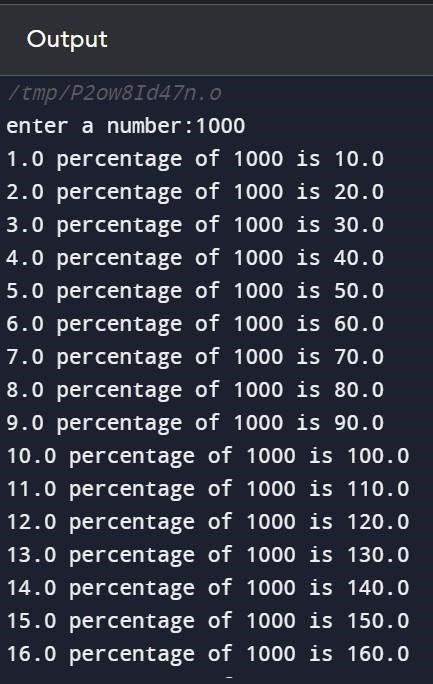
} int main(){

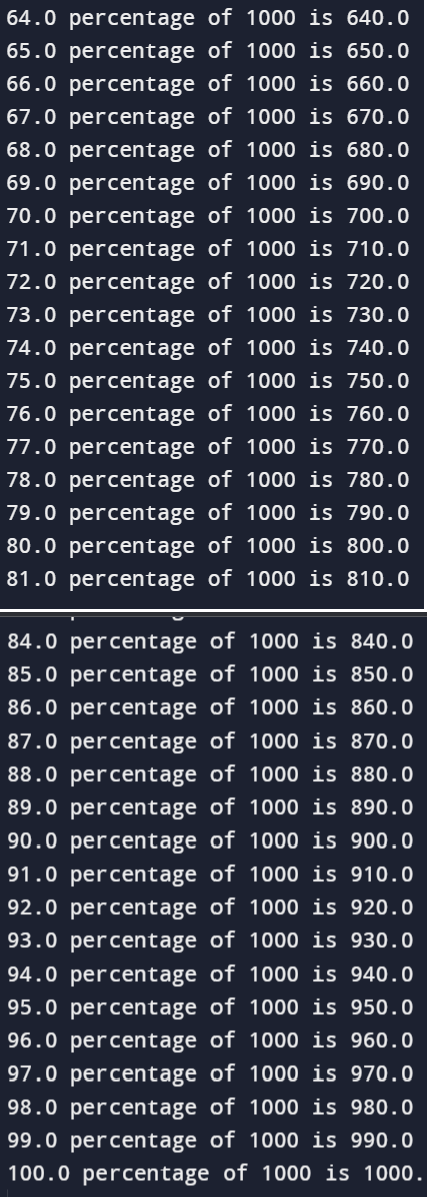
int n; printf("enter a number:");

scanf("%d",&n); float i=1; percentage(n,i);

return 0;

}





**Observation:** in this program there is a float function called percentage which returns the value percentage from one to 100 ,till it reaches the base case when the index is greater than 100. It prints the percentage of the inputted number from 1 to 100.

**d) C program to find Sum of digits of a number program using recursion.** #include <stdio.h> int sumofdigits(int n){ if(n==0){ return 0;

}

return (n%10)+sumofdigits(n/10);

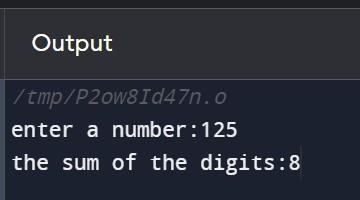
} int main(){ int n;

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

printf("the sum of the digits:%d",sumofdigits(n));

return 0;

}



**Observation:** in this program the sum of digits function is a recursive function that calculates the sum of digits breaks down the problem of finding the last digit and adding it to the sum of the remaining digits and the process continues till it reaches the base case when it is reached 0.

**e) C program to pass two-dimensional array (Two-D array) to a function** #include <stdio.h>

void displayarr(int a,int b,int arr[a][b]){ printf("The 2D array:\n"); for (int i = 0; i <a; i++) { for (int j = 0; j <b; j++) { printf("%d\t", arr[i][j]);

}

printf("\n");

}

}

int main(){ int a,b;

printf("enter no of rows and columns:");

scanf("%d %d",&a,&b);

int arr[a][b];

printf("Enter elements of the 2D array:\n");

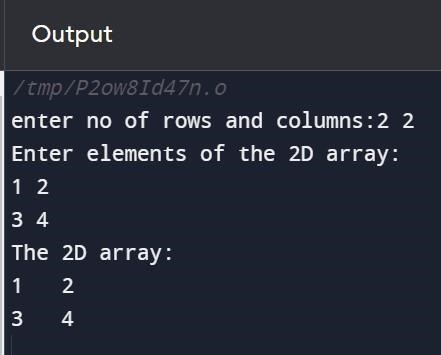
for (int i = 0; i <a; i++) { for (int j = 0; j <b; j++) { scanf("%d", &arr[i][j]);

}

}

displayarr(a,b,arr); return 0;

}



**Observation:** in this program net display function takes the parameters open number of rows number of columns and the 2D array. In the function elements of the 2D array using two for loops.

**f) C program to find the LCM (Lowest Common Multiple) of given numbers using recursion** #include <stdio.h> int gcd(int num1, int num2) { if (num2 == 0) { return num1;

} else { return gcd(num2, num1 % num2); } }

int lcm(int num1, int num2) { return (num1 \* num2) / gcd(num1, num2);

}

// Function to find the LCM of an array of numbers

int lcmOfArray(int arr[], int n) { if (n == 1) { return arr[0];

} else { return lcm(arr[n - 1], lcmOfArray(arr, n - 1)); }

}

int main() { int n;

printf("Enter the number of elements in the array: "); scanf("%d", &n);

if (n <= 0) { printf("Please enter a valid number of elements.\n"); return 1; // Return an error code

}

int numbers[n]; for (int i = 0; i < n; i++) {

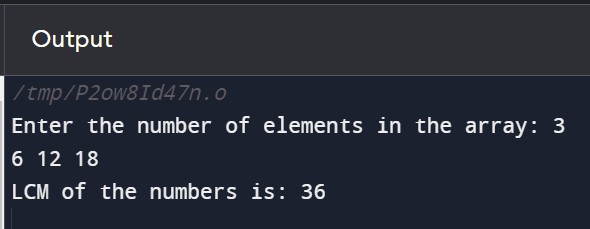
scanf("%d", &numbers[i]);

}

int result = lcmOfArray(numbers, n); printf("LCM of the numbers is: %d\n", result);

return 0;

}



**Observation:** this function uses the LCM function to find the LCM of the array elements recursivvely. Nielsen function calculates the LCM of two numbers using their greatest common divisor.

|  |  |
| --- | --- |
| **LAB 8**        **a) C program to read and print an Employee’s Details using Structure** #include <stdio.h>    // Define a structure for employee details  struct Employee { char name[50]; int employeeID; float salary;  };  int main() {  // Declare a structure variable for employee struct Employee employee1;    // Input employee details printf("Enter employee name: "); scanf("%s", employee1.name);    printf("Enter employee ID: ");  scanf("%d", &employee1.employeeID);    printf("Enter employee salary: "); scanf("%f", &employee1.salary);    // Print employee details printf("\nEmployee Details:\n"); printf("Name: %s\n", employee1.name);  printf("Employee ID: %d\n", employee1.employeeID); printf("Salary: %.2f\n", employee1.salary);    return 0;  } | 65 |

|  |
| --- |
| 66    **Observation:**this program uses the structure variable to store and retrieve employee details without the need for a pointer . the structure itself serves as a cointainer for related information about the employee.      **b) C program to demonstrate example of structure of array** #include <stdio.h>    // Define a structure for student details struct Student { char name[50]; int rollNumber; float marks;  };  int main() {  // Declare an array of structures for multiple students struct Student students[2];    // Input details for each student in the array for (int i = 0; i < 2; i++) { printf("Enter details for student %d:\n", i + 1);    printf("Enter name: ");  scanf("%s", students[i].name);    printf("Enter roll number: ");  scanf("%d", &students[i].rollNumber);    printf("Enter marks: "); scanf("%f", &students[i].marks);    printf("\n");  } |

6

7

//

Print

details

of

each stu

dent

in

the

array

printf("Student

Details:

\

n");

for (int i = 0; i < 2; i++) {

printf("Student

%d:

\

n",

i

+

1)

;

printf("Name:

%s

\

n",

students[i].name);

printf("Roll

Number:

%d

\

n",

students[i].rollNumber);

printf("Marks: %.2f

\

n", students[i].marks);

printf("

\

n");

}

return

;

0

}

**Observation:**

in this program,struct student defines a structure with three members ,name,rollno

and marks.struct student

students[2] declares an array of structures to store details for multiple

students.

And

finally using

the

for loop the

details

are

printed

out.

**c)**

**C**

**program**

**to**

**demonstrate**

**example**

**of**

**Nested**

**Structure**

#include

<

stdio.h

>

//

Define

a

structure

for

address

details

struct

Address

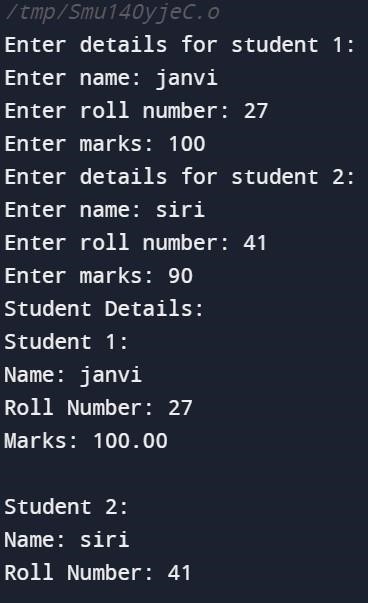
{

char

street[50];

char

city[50];



|  |  |
| --- | --- |
| char state[50];  };    // Define a structure for employee details with a nested structure for address struct Employee { char name[50]; int employeeID;  float salary;  struct Address empAddress;  };  int main() {  // Declare an employee structure variable struct Employee employee1;    // Input details for the employee and address printf("Enter employee details:\n"); printf("Enter name: "); scanf("%s", employee1.name);    printf("Enter employee ID: ");  scanf("%d", &employee1.employeeID);    printf("Enter salary: ");  scanf("%f", &employee1.salary);  printf("Enter address details:\n"); printf("Enter street: ");  scanf("%s", employee1.empAddress.street);    printf("Enter city: ");  scanf("%s", employee1.empAddress.city);    printf("Enter state: ");  scanf("%s", employee1.empAddress.state);    // Print employee details along with nested address details printf("\nEmployee Details:\n"); printf("Name: %s\n", employee1.name);  printf("Employee ID: %d\n", employee1.employeeID);  printf("Salary: %.2f\n", employee1.salary); printf("Address:\n");  printf("Street: %s\n", employee1.empAddress.street); | 68 |

|  |
| --- |
| 69  printf("City: %s\n", employee1.empAddress.city); printf("State: %s\n", employee1.empAddress.state);    return 0;  }      **Observation:**the struct address defines a structurer for address details members street city and state.struct employee defines a structure for employing details with members name employee ID salary and nested structure.the program declares an employee1 structure variable and uses nested structures to store details.        **d) C program to demonstrate example structure pointer using user define function** #include <stdio.h>    // Define a structure for book details struct Book { char title[100]; int year; float price;  };    // Function to input book details using structure pointer  void inputBookDetails(struct Book \*book) { printf("Enter book title: "); scanf("%s", book->title);    printf("Enter publication year: "); scanf("%d", &book->year); |

printf("Enter price: "); scanf("%f", &book->price);

}

// Function to display book details using structure pointer void displayBookDetails(const struct Book \*book) { printf("\nBook Details:\n"); printf("Title: %s\n", book->title);

printf("Year of publication: %d\n", book->year); printf("Price: %.2f\n", book->price);

}

int main() {

// Declare a book structure variable and a pointer to the structure struct Book myBook;

struct Book \*ptrBook;

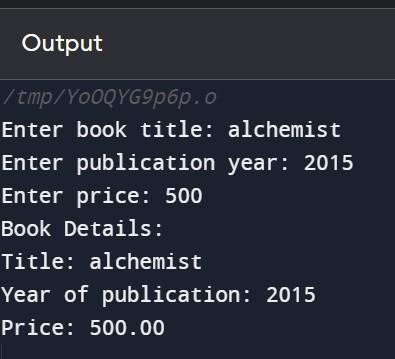
// Initialize the pointer to the address of the structure variable ptrBook = &myBook;

// Input book details using the function with structure pointer inputBookDetails(ptrBook);

// Display book details using the function with structure pointer displayBookDetails(ptrBook);

return 0;

}



**Observation:**struct book defines a structurer for book details with members title,year,and price the program declares a mybook structure variable and a ptrbook pointer to the structure .the inputbookdetails function takes a structure pointer as a parameter and input book details using

|  |  |
| --- | --- |
| that pointer.and then the displaybookdetails function takes a counst structure pointer as a paramenter and displays book details using that pointer .        **e) C program to add two distances in feet and inches using structure** #include <stdio.h>    // Define a structure for distance struct Distance { int feet; float inches;  };    // Function to add two distances  struct Distance addDistances(struct Distance d1, struct Distance d2) { struct Distance result;  result.feet = d1.feet + d2.feet;  result.inches = d1.inches + d2.inches;    // If the sum of inches is greater than or equal to 12, convert to feet and update inches if (result.inches >= 12.0) { result.feet += (int)(result.inches / 12.0); result.inches = fmod(result.inches, 12.0); }    return result;  }    int main() {  // Declare two distance structure variables  struct Distance distance1, distance2, sumDistance;    // Input distance1 details printf("Enter distance 1:\n"); printf("Feet: ");  scanf("%d", &distance1.feet); printf("Inches: ");  scanf("%f", &distance1.inches);    // Input distance2 details printf("\nEnter distance 2:\n"); | 71 |

printf("Feet: ");

scanf("%d", &distance2.feet); printf("Inches: ");

scanf("%f", &distance2.inches);

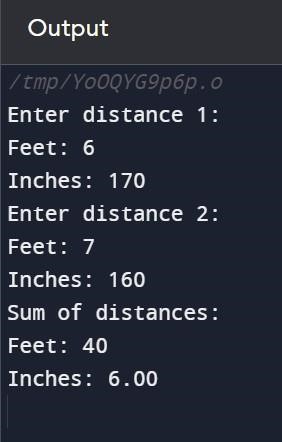
// Add distances using the function

sumDistance = addDistances(distance1, distance2);

// Display the result printf("\nSum of distances:\n"); printf("Feet: %d\n", sumDistance.feet); printf("Inches: %.2f\n", sumDistance.inches);

return 0;

}



**Observation:**the struct distance defines a structure for distance with members feet and inches .the adddistances function takes two distance structures as parameters and returns the sum of the distances . the adddistance function is called to calculate the sum of the distances .the result is then displayed.

## LAB 9

**a) C program to read array elements and print the value with the addresses**

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

printf("Enter the number of elements in the array: "); scanf("%d", &n);

if (n <= 0) { printf("Please enter a valid number of elements.\n"); return 1; // Return an error code

}

int myArray[n]; printf("Enter array elements:\n"); for (int i = 0; i < n; i++) { scanf("%d", &myArray[i]);

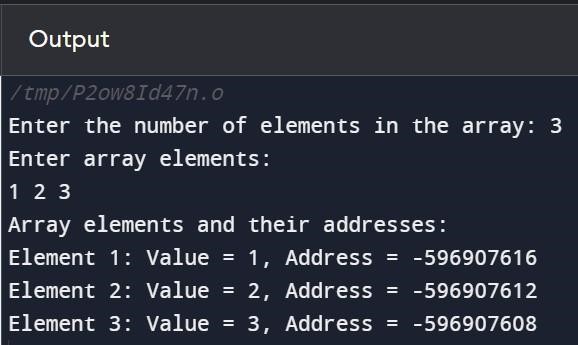
}

// Print array elements with addresses printf("\nArray elements and their addresses:\n");

for (int i = 0; i < n; i++) { printf("Element %d: Value = %d, Address = %d\n", i + 1, myArray[i],&myArray[i]); }

return 0;

}



**Observation:** the user is prompted to enter the number of elements in the array and then did the element in the area then the program prints the values of the array elements along with the addresses of the elements using the & symbol.

**b) C program to count vowels and consonants in a string using pointer**

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

void countVowelsConsonants(char \*str, int \*vowelCount, int \*consonantCount) { while (\*str) { char currentChar = tolower(\*str);

if (currentChar >= 'a' && currentChar <= 'z') { if (currentChar == 'a' || currentChar == 'e' || currentChar == 'i' || currentChar == 'o' || currentChar == 'u') {

(\*vowelCount)++;

} else {

(\*consonantCount)++;

}

}

str++;

}

}

int main() { char str[100];

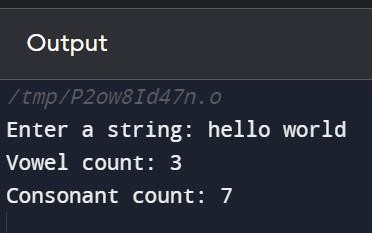
int vowelCount = 0, consonantCount = 0;

printf("Enter a string: "); fgets(str, sizeof(str), stdin);

countVowelsConsonants(str, &vowelCount, &consonantCount); printf("Vowel count: %d\n", vowelCount); printf("Consonant count: %d\n", consonantCount);

return 0;

}



**Observation:** The countvowelsconsonents function takes a string pointer and two integer pointers as the parameters. It iterates through each character of the string converts it to lowercase using two lower functions and checks if it's vowel or consonant ,depending on that increase the respective variable. And then in the main function the vowel count and the consonant count are given as output.

**c) C program to demonstrate example of double pointer** #include <stdio.h>

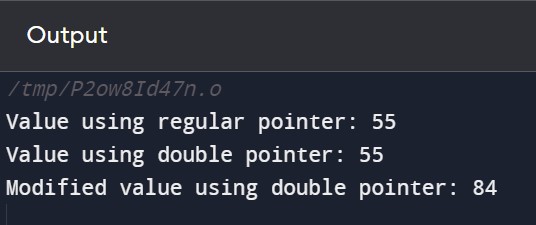
int main() { int n = 55; int \*ptr = &n; int \*\*pptr = &ptr; printf("Value using regular pointer: %d\n", \*ptr); printf("Value using double pointer: %d\n", \*\*pptr);

\*ptr = 84;

// Displaying the modified value using double pointer printf("Modified value using double pointer: %d\n", \*\*pptr);

return 0;

}



**Observation:** first we declare an integer variable n to 55. And then using \*ptr is equal to &n declares A pointer to an integer and initialized with the address of that number and then we use \*\*pptr declares a double pointer to an integer pointer and initiates it with the address of ptr and then modify the value of the number using pointer. And then display the modifier value using the double pointer.

|  |  |
| --- | --- |
| **d) C program to read and print student details using structure pointer** #include <stdio.h>    // Define a structure for student details struct Student { char name[50]; int rollNumber; float marks; };  int main() {  // Declare a structure variable and a pointer to the structure struct Student student1; struct Student \*ptrStudent;    // Initialize the pointer to the address of the structure variable ptrStudent = &student1;    // Input student details using the pointer printf("Enter student name: "); scanf("%s", ptrStudent->name);    printf("Enter roll number: ");  scanf("%d", &ptrStudent->rollNumber);    printf("Enter marks: ");  scanf("%f", &ptrStudent->marks);    // Print student details using the pointer printf("\nStudent Details:\n");  printf("Name: %s\n", ptrStudent->name);  printf("Roll Number: %d\n", ptrStudent->rollNumber); printf("Marks: %.2f\n", ptrStudent->marks);    return 0;  } | 76 |

7

7

**Observation:**

struct student defines three members name row number and marks.struct student1

declares a structure variable.ptrstudent =&student1

initialize the pointer with the address of the

structure

variable.

Using the

pointer

student

details

are

input

from

the

user and

printed

as

output.

**e)**

**C**

**program**

**to**

**swap**

**two numbers**

**using**

**pointers**

#include

<

stdio.h

>

void

swap(int

\*a,int

\*b){

int

temp=\*a;

\*a=

\*b;

\*b=temp;

}

int main(){

int

a,b;

printf("enter

the

values

of

a

and

b:");

scanf("%d %d",&a,&b);

swap(&a,&b);

printf("a:%d

b:%d",a,b);

return

;

0

}

**Observation:**

the swap function take two integers pointers int\*a and int \*b as the parameters and

the values of A&B are swapped using a temporary variable . The function is called in the main

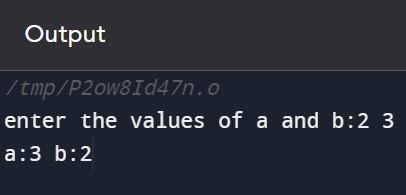
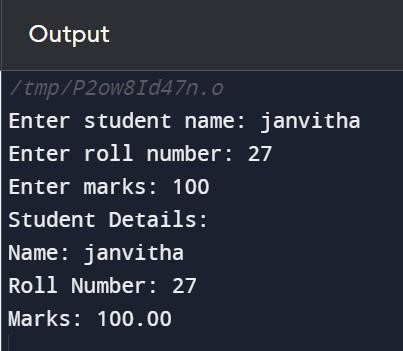
function

to

print

A

and B values.



|  |  |
| --- | --- |
| **LAB 10**        **a) C program to input and print text using Dynamic Memory Allocation.** #include <stdio.h>  #include <stdlib.h>    int main() { char \*text; int length;    // Get the length of the text from the user printf("Enter the length of the text: "); scanf("%d", &length);    // Allocate memory for the text  text = (char \*)malloc((length + 1) \* sizeof(char)); // +1 for the null terminator    if (text == NULL) { printf("Memory allocation failed.\n"); return 1; // Return an error code  }    // Get text from the user printf("Enter the text: "); scanf(" %[^\n]", text);    // Print the entered text  printf("Entered text: %s\n", text);    // Free the allocated memory free(text);    return 0; // Return success } | 78 |

|  |  |  |
| --- | --- | --- |
| **Observation**: This program prompts the user to enter the length of the text, dynamically   |  | | --- | | free() |   allocates memory for the text, takes the text as input, prints it, and then frees the allocated memory using . Make sure to handle memory allocation failure appropriately      **b) C program to read and print the student details using structure and Dynamic Memory Allocation**  #include <stdio.h>  #include <stdlib.h>    // Define a structure for student details struct Student { char name[50]; int rollNumber; float marks;  };    int main() { int numStudents;  struct Student \*students;    // Get the number of students from the user printf("Enter the number of students: "); scanf("%d", &numStudents);    // Dynamically allocate memory for an array of students  students = (struct Student \*)malloc(numStudents \* sizeof(struct Student));    if (students == NULL) { printf("Memory allocation failed.\n"); return 1; // Return an error code }    // Input details for each student for (int i = 0; i < numStudents; ++i) { printf("\nEnter details for student %d:\n", i + 1);    printf("Name: ");  scanf(" %[^\n]", students[i].name);    printf("Roll Number: ");  scanf("%d", &students[i].rollNumber); | 79 |

printf("Marks: ");

scanf("%f", &students[i].marks); }

// Print details for each student printf("\nStudent Details:\n"); for (int i = 0; i < numStudents; ++i) { printf("\nDetails for student %d:\n", i + 1);

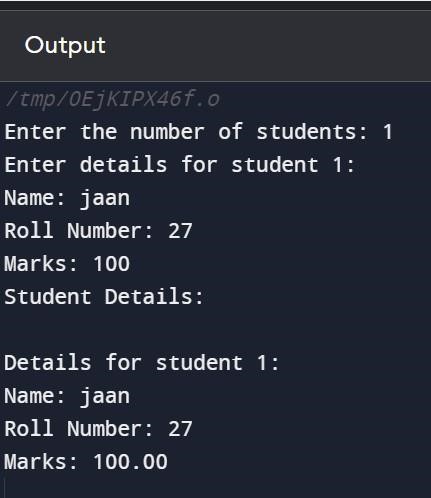
printf("Name: %s\n", students[i].name);

printf("Roll Number: %d\n", students[i].rollNumber); printf("Marks: %.2f\n", students[i].marks); }

// Free the allocated memory free(students);

return 0; // Return success

}



**Observation:** This program uses a structure Student to store details such as name, roll number, and marks for each student. It dynamically allocates memory for an array of students, reads the details for each student, prints the details, and then frees the allocated memory.

**c) C program to create memory for int, char and float variable at run time.** #include <stdio.h>

#include <stdlib.h>

int main() { int \*intVariable;

|  |  |
| --- | --- |
| char \*charVariable; float \*floatVariable;    // Allocate memory for int  intVariable = (int \*)malloc(sizeof(int));    // Check for memory allocation failure if (intVariable == NULL) { printf("Memory allocation failed for int variable.\n"); return 1; // Return an error code  }    // Allocate memory for char  charVariable = (char \*)malloc(sizeof(char));    // Check for memory allocation failure  if (charVariable == NULL) { printf("Memory allocation failed for char variable.\n");    // Free the previously allocated memory free(intVariable);  return 1; // Return an error code }    // Allocate memory for float  floatVariable = (float \*)malloc(sizeof(float));    // Check for memory allocation failure if (floatVariable == NULL) { printf("Memory allocation failed for float variable.\n");    // Free the previously allocated memory free(intVariable); free(charVariable);  return 1; // Return an error code }    // Input values for the variables printf("Enter an integer: "); scanf("%d", intVariable);  printf("Enter a character: "); scanf(" %c", charVariable); | 81 |

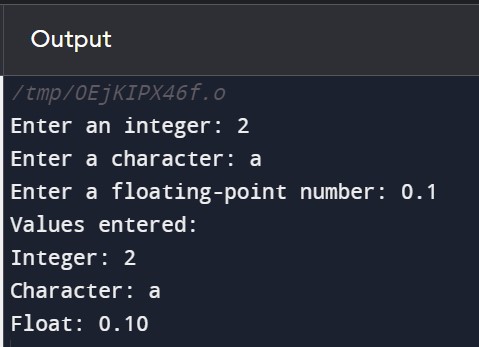
printf("Enter a floating-point number: "); scanf("%f", floatVariable);

// Print the values of the variables printf("\nValues entered:\n"); printf("Integer: %d\n", \*intVariable); printf("Character: %c\n", \*charVariable); printf("Float: %.2f\n", \*floatVariable);

// Free the allocated memory free(intVariable); free(charVariable); free(floatVariable);

return 0; // Return success

}



|  |  |  |
| --- | --- | --- |
| int | , a | char |

**Observation:** This program dynamically allocates memory for an , and a float

variable, takes user input for each variable, prints the entered values, and then frees the allocated memory. Make sure to handle memory allocation failure appropriately.

**d) program to read a one-dimensional array, print sum of all elements along with inputted array elements using Dynamic Memory Allocation.**

#include <stdio.h>

#include <stdlib.h>

int main() { int \*array;

int size, sum = 0;

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

|  |  |
| --- | --- |
| // Dynamically allocate memory for the array array = (int \*)malloc(size \* sizeof(int));    // Check for memory allocation failure if (array == NULL) { printf("Memory allocation failed.\n"); return 1; // Return an error code  }    // Input array elements from the user printf("Enter the array elements:\n");  for (int i = 0; i < size; ++i) { printf("Element %d: ", i + 1); scanf("%d", &array[i]); sum += array[i];  }    // Print the sum of array elements  printf("\nSum of array elements: %d\n", sum);    // Print the inputted array elements printf("\nInputted array elements:\n");  for (int i = 0; i < size; ++i) { printf("%d ", array[i]);  }    // Free the allocated memory free(array);    return 0; // Return success  } | 83 |

**Observation:** This program dynamically allocates memory for an array, takes user input for each array element, calculates the sum of all elements, prints the sum, and then prints the inputted array elements. Make sure to handle memory allocation failure appropriately.

## LAB 11

**a) C program to create a text file using file handling** #include <stdio.h>

int main() {

FILE \*filePointer; // Declare a file pointer

// Open the file in write mode ("w") filePointer = fopen("hello.txt", "w");

// Check if the file was opened successfully if (filePointer == NULL) { printf("Error opening the file.\n"); return 1; // Return an error code }

// Write content to the file

fprintf(filePointer, "Hello,world!!.\n");

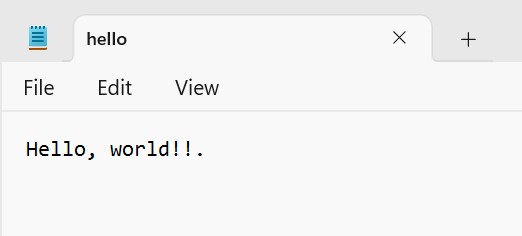
// Close the file fclose(filePointer);

printf("File created successfully.\n");

return 0; // Return success

}

|  |
| --- |
| fopen |

**Observation**: the function is used to open a file called "hello.txt" in write mode ("w"). The fprintf function is then used to write content to the file, and finally, the fclose function is used to close the file.

**b) C program to find number of lines in a file**

#include <stdio.h>

int main() {

FILE \*filePointer; char filename[100]; char ch;

int lineCount = 0;

// Get the filename from the user printf("Enter the filename: "); scanf("%s", filename);

// Open the file in read mode filePointer = fopen(filename, "r");

// Check if the file exists if (filePointer == NULL) { printf("File not found or cannot be opened.\n"); return 1; // Return an error code

}

// Count the number of lines

while ((ch = fgetc(filePointer)) != EOF) { if (ch == '\n') { lineCount++;

}

}

// Close the file

fclose(filePointer);

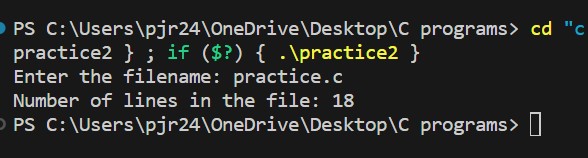
// Print the result

printf("Number of lines in the file: %d\n", lineCount);

return 0; // Return success

**Observation:** This program prompts the user to enter the filename, opens the file in read mode, and then counts the number of newline characters ('\n') to determine the number of lines in the file. Finally, it prints the result.

}



**c) C program to rename a file using the rename () function** #include <stdio.h>

int main() { char oldFileName[100], newFileName[100];

// Get the current filename from the user printf("Enter the current filename: ");

scanf("%s", oldFileName);

// Get the new filename from the user printf("Enter the new filename: "); scanf("%s", newFileName);

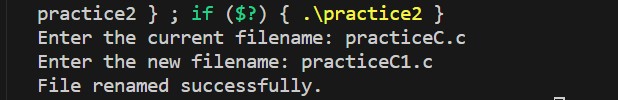
// Rename the file

if (rename(oldFileName, newFileName) == 0) { printf("File renamed successfully.\n");

} else { perror("Error renaming file"); return 1; // Return an error code }

return 0; // Return success

}



**Observation:** This program prompts the user to enter the current filename and the new filename. It then uses the rename() function to rename the file. If the renaming is successful, it prints a success message. Otherwise, it prints an error message using perror() and returns an error code.

**d) C program to create, write and read text in/from file** #include <stdio.h>

int main() {

FILE \*filePointer; char fileName[100]; char text[100];

|  |  |
| --- | --- |
| // Get the filename from the user printf("Enter the filename: "); scanf("%s", fileName);    // Create a file and write text to it filePointer = fopen(fileName, "w");    if (filePointer == NULL) { printf("Error creating file.\n"); return 1; // Return an error code  }    // Get text from the user to write to the file printf("Enter text to write to the file: "); scanf(" %[^\n]", text);  fprintf(filePointer, "%s", text);    // Close the file fclose(filePointer);    // Open the file for reading  filePointer = fopen(fileName, "r");    if (filePointer == NULL) { printf("Error opening file for reading.\n"); return 1; // Return an error code }    // Read and display the content of the file printf("\nContents of the file:\n");    while (fscanf(filePointer, "%s", text) != EOF) { printf("%s ", text);  }    // Close the file fclose(filePointer);    return 0; // Return success } | 88 |

8

9

**Observation:**

This

program prompts

the

user to

enter

a

filename,

creates

a

file

with

that name,

writes

text to it, and then reads and displays the content of the file. Make sure to handle errors

appropriately,

especially when

dealing with

file

operations

.

