**1.Program to swap two numbers without using third variable**

**Solution:-**

#include<stdio.h>

int main() {

double first, second, temp;

printf("Enter first number: ");

scanf("%lf", &first);

printf("Enter second number: ");

scanf("%lf", &second);

temp = first;

first = second;

second = temp;

printf("\nAfter swapping, first number = %.2lf\n", first);

printf("After swapping, second number = %.2lf", second);

return 0;

}

**2. Add two numbers represented by linked lists**

**Solution:-**

#include <stdio.h>

#include <stdlib.h>

typedef struct Node {

int data;

struct Node\* next;

}Node;

Node\* newNode(int data)

{

Node\* new\_node = (Node \*)malloc(sizeof(Node));

new\_node->data = data;

new\_node->next = NULL;

return new\_node;

}

void push(Node\*\* head\_ref, int new\_data)

{

Node\* new\_node = newNode(new\_data);

new\_node->next = (\*head\_ref);

(\*head\_ref) = new\_node;

}

Node\* addTwoLists(Node\* first, Node\* second)

{

Node\* res = NULL;

Node \*temp, \*prev = NULL;

int carry = 0, sum;

while (first != NULL || second != NULL) {

sum = carry + (first ? first->data : 0) + (second ? second->data : 0);

carry = (sum >= 10) ? 1 : 0;

sum = sum % 10;

temp = newNode(sum);

if (res == NULL)

res = temp;

else

prev->next = temp;

prev = temp;

if (first)

first = first->next;

if (second)

second = second->next;

}

if (carry > 0)

temp->next = newNode(carry);

return res;

}

Node\* reverse(Node\* head)

{

if (head == NULL || head->next == NULL)

return head;

Node\* rest = reverse(head->next);

head->next->next = head;

head->next = NULL;

return rest;

}

void printList(Node\* node)

{

while (node != NULL) {

printf("%d ",node->data);

node = node->next;

}

printf("\n");

}

int main(void)

{

Node\* res = NULL;

Node\* first = NULL;

Node\* second = NULL;

push(&first, 6);

push(&first, 4);

push(&first, 9);

push(&first, 5);

push(&first, 7);

printf("First list is ");

printList(first);

push(&second, 4);

push(&second, 8);

printf("Second list is ");

printList(second);

first = reverse(first);

second = reverse(second);

res = addTwoLists(first, second);

res = reverse(res);

printf("Resultant list is ");

printList(res);

return 0;

}

**3. Find minimum number of currency notes and values that sum to given amount**

**Solution:-**

#include <stdio.h>

int main()

{

int a[8]={500,100,50,20,10,5,2,1},m,temp,i;

printf("Enter the amount:");

scanf("%d",&m);

temp=m;

for(i=0;i<8;i++)

{

printf("\n%d notes is:%d",a[i],temp/a[i]);

temp=temp%a[i];

}

}

4**. Program to find line passing through 2 Points**

**Solution:-**

#include <stdio.h>

#include <math.h>

void main()

{

float slope, intercept;

float x1, y1, x2, y2;

float dx, dy;

printf("Program to find the equation of a line given two end points\n");

printf("Enter X1: ");

scanf("%f", &x1);

printf("Enter Y1: ");

scanf("%f", &y1);

printf("Enter X2: ");

scanf("%f", &x2);

printf("Enter Y2: ");

scanf("%f", &y2);

dx = x2 - x1;

dy = y2 - y1;

slope = dy / dx;

intercept = y1 - slope \* x1;

printf("Equation of the line with end points (%.2f, %.2f) and (%.2f, %.2f) : Y = %.2fX %c %.2f\n", x1, y1, x2, y2, slope, (intercept < 0) ? ' ' : '+', intercept);

}

**5. Program for array rotation**

**Solution:**

#include <stdio.h>

int main()

{

int arr[] = {1, 2, 3, 4, 5};

int length = sizeof(arr)/sizeof(arr[0]);

int n = 3;

printf("Original array: \n");

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

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

}

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

int j, last;

last = arr[length-1];

for(j = length-1; j > 0; j--){

arr[j] = arr[j-1];

}

arr[0] = last;

}

printf("\n");

printf("Array after right rotation: \n");

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

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

}

return 0;

}

**6. Find the longest substring with k unique characters in a given string**

Solution:-

#include <bits/stdc++.h>

using namespace std;

void longestKSubstr(string s, int k)

{

int i = 0;

int j = 0;

map<char, int> m;

int maxl = -1;

int n = s.size();

while (i < n) {

j = i;

while (j < n) {

char curchar = s[j];

m[curchar] += 1;

if (m.size() == k) {

int curlength = j - i + 1;

maxl = max(maxl, curlength);

}

if (m.size() > k) {

break;

}

j++;

}

i++;

}

if (maxl == -1) {

cout << "Not enough unique characters";

}

else {

cout << maxl;

}

}

int main()

{

string s = "aabacbebebe";

int k = 3;

longestKSubstr(s, k);

return 0;

}

**7. Find the longest substring with k unique characters in a given string**

**Solution:-**

#include <iostream>

using namespace std;

bool isEven(int n) { return (n % 2 == 0); }

int main()

{

int n = 101;

isEven(n) ? cout << "Even" : cout << "Odd";

return 0;

}

**8. C/C++ program to make a simple calculator**

**Solution:-**

#include <stdio.h>

int main() {

char op;

double first, second;

printf("Enter an operator (+, -, \*, /): ");

scanf("%c", &op);

printf("Enter two operands: ");

scanf("%lf %lf", &first, &second);

switch (op) {

case '+':

printf("%.1lf + %.1lf = %.1lf", first, second, first + second);

break;

case '-':

printf("%.1lf - %.1lf = %.1lf", first, second, first - second);

break;

case '\*':

printf("%.1lf \* %.1lf = %.1lf", first, second, first \* second);

break;

case '/':

printf("%.1lf / %.1lf = %.1lf", first, second, first / second);

break;

default:

printf("Error! operator is not correct");

}

return 0;

}

**9. Prime Numbers**

Solution:-

#include<stdio.h>

int main(){

int n,i,m=0,flag=0;

printf("Enter the number to check prime:");

scanf("%d",&n);

m=n/2;

for(i=2;i<=m;i++)

{

if(n%i==0)

{

printf("Number is not prime");

flag=1;

break;

}

}

if(flag==0)

printf("Number is prime");

return 0;

}

10**. Program for Fibonacci numbers**

**Solution:-**

#include<stdio.h>

int main()

{

int n1=0,n2=1,n3,i,number;

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

scanf("%d",&number);

printf("\n%d %d",n1,n2);

for(i=2;i<number;++i)

{

n3=n1+n2;

printf(" %d",n3);

n1=n2;

n2=n3;

}

return 0;

}

11. **Palindrome number program**

**Solution:-**

#include<stdio.h>

int main()

{

int n,r,sum=0,temp;

printf("enter the number=");

scanf("%d",&n);

temp=n;

while(n>0)

{

r=n%10;

sum=(sum\*10)+r;

n=n/10;

}

if(temp==sum)

printf("palindrome number ");

else

printf("not palindrome");

return 0;

}

12. **Min and Max in an array program**

**Solution:-**

#include <stdio.h>

#define MAX\_SIZE 100

int main()

{

int arr[MAX\_SIZE];

int i, max, min, size;

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

scanf("%d", &size);

printf("Enter elements in the array: ");

for(i=0; i<size; i++)

{

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

}

max = arr[0];

min = arr[0];

for(i=1; i<size; i++)

{

if(arr[i] > max)

{

max = arr[i];

}

if(arr[i] < min)

{

min = arr[i];

}

}

printf("Maximum element = %d\n", max);

printf("Minimum element = %d", min);

return 0;

}

**13. Average of an array program**

**Solution:-**

#include <stdio.h>

int main() {

int n, i;

float num[100], sum = 0.0, avg;

printf("Enter the numbers of elements: ");

scanf("%d", &n);

while (n > 100 || n < 1) {

printf("Error! number should in range of (1 to 100).\n");

printf("Enter the number again: ");

scanf("%d", &n);

}

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

printf("%d. Enter number: ", i + 1);

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

sum += num[i];

}

avg = sum / n;

printf("Average = %.2f", avg);

return 0;

}

**14. Reverse an array program**

**Solution:-**

#include <stdio.h>

#include <stdlib.h>

#define n 6

int main(){

int arr[n] = {9, 8, 7, 2, 4, 3};

int temp;

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

temp = arr[i];

arr[i] = arr[n-i-1];

arr[n-i-1] = temp;

}

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

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

}

}

**15.Sum of two numbers**

**Solution:-**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int nums[100],i,j,size,target,sum=0;

printf("Enter the size of an array\n");

scanf("%d",&size);

printf("Enter an array elements\n");

for(i=0;i<size;i++)

{

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

}

printf("Enter the value for target\n");

scanf("%d",&target);

for(i=0;i<size;i++)

{

for(j=i+1;j<size;j++)

{

sum=nums[i]+nums[j];

if(sum==target)

{

printf("Output : [%d, %d]",i,j);

}

}

}

return 0;

}

**16. Check if array is palindrome program**

**Solution:-**

#include <stdio.h>

int pallindrome(int arr[], int n) {

int i, j, flag = 0;

for(i = 0, j=n-1; i< n/2, j>=n/2; i++, j--) {

if(arr[i]!=arr[j]) {

flag = 1;

break;

}

}

if (flag == 1)

return 0;

else

return 1;

}

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

int arr[] = {1, 0, 2, 3, 2, 2, 1};

int n = sizeof(arr)/sizeof(arr[0]);

if(pallindrome(arr, n)) {

printf("Array is pallindrome\n");

}

else

printf("Array is not pallindrome\n");

return 0;

}

**17. Frequency of an element in an array program**

**Solution:-**#include <stdio.h>

int main()

{

int arr[] = {1, 2, 8, 3, 2, 2, 2, 5, 1};

int length = sizeof(arr)/sizeof(arr[0]);

int fr[length];

int visited = -1;

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

int count = 1;

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

if(arr[i] == arr[j]){

count++;

fr[j] = visited;

}

}

if(fr[i] != visited)

fr[i] = count;

}

printf("---------------------\n"); printf(" Element | Frequency\n");

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

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

if(fr[i] != visited){

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

printf(" | ");

printf(" %d\n", fr[i]);

}

}

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

return 0;

}

**18. Find non repeating element in an array program**

**Solution:-**

#include <stdio.h>

int main()

{

int arr[] = {21, 30, 10, 2, 10, 20, 30, 11};

int n = sizeof(arr)/sizeof(arr[0]);

int visited[n];

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

if(visited[i]==0){

int count = 1;

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

if(arr[i]==arr[j]){

count++;

visited[j]=1;

}

}

if(count==1)

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

} }return 0;}

**19.Merge sort**

**Solution:-**

#include<stdlib.h>

#include<stdio.h>

// Merge Function

void merge(int arr[], int l, int m, int r)

{

int i, j, k;

int n1 = m - l + 1;

int n2 = r - m;

int L[n1], R[n2];

for (i = 0; i < n1; i++)

L[i] = arr[l + i];

for (j = 0; j < n2; j++)

R[j] = arr[m + 1+ j];

i = 0;

j = 0;

k = l;

while (i < n1 && j < n2)

{

if (L[i] <= R[j])

{

arr[k] = L[i];

i++;

}

else

{

arr[k] = R[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = L[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = R[j];

j++;

k++;

}

}

**20. Majority element program**

**Solution:-**

#include <stdio.h>

#define ARRAY\_SIZE 100

void getMajorityElement(int \*array, int size) {

int i, majorityIndex = 0, count = 1;

for(i = 1; i < size; i++) {

if(array[majorityIndex] == array[i])

count++;

else

count--;

if(count == 0) {

majorityIndex = i;

count = 1;

}

}

count = 0;

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

if(array[i] == array[majorityIndex])

count++;

}

if(count > (size/2))

printf("Majority Element : %d\n", array[majorityIndex]);

else

printf("No Majority Element Found\n");

}

int main(){

int i, array[ARRAY\_SIZE], count, sum;

printf("Enter the number of elements in Array\n");

scanf("%d", &count);

printf("Enter %d numbers\n", count);

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

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

}

getMajorityElement(array, count);

return 0;

}

**21. Average salary excluding minimum and maximum salary program**

**Solution:-**

#include <bits/stdc++.h>

using namespace std;

double average(vector<int>& salary) {

int n=salary.size();

int mn=INT\_MAX,mx=INT\_MIN,sum=0;

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

{

sum=sum+salary[i];

mn=min(mn,salary[i]);

mx=max(mx,salary[i]);

}

double ans=(sum-mn-mx)/((n-2)\*1.0);

if(n>2)

return ans;

else

return 0;

}

int main()

{

vector<int> arr = {8000,9000,2000,3000,6000,1000};

cout<<average(arr)<<endl;

return 0;

}

**22. Sort by parity program**

**Solution:-**

# include <stdio.h>

# define bool int

bool getParity(unsigned int n)

{

bool parity = 0;

while (n)

{

parity = !parity;

n = n & (n - 1);

}

return parity;

}int main()

{

unsigned int n = 7;

printf("Parity of no %d = %s", n,

(getParity(n)? "odd": "even"));

getchar();

return 0;

}

**23. Sqrt of number program**

**Solution:-**

#include <math.h>

#include <stdio.h>

double findSQRT(double N) { return sqrt(N); }

int main()

{

int N = 12;

printf("%f ", findSQRT(N));

return 0;

}

**24. Peak index in a mountain array**

**Solution:-**

**#include <iostream>**

**using namespace std;**

**int peakIndex(int arr[],int high)**

**{**

**int low=0;**

**int mid;**

**high-=1;**

**while( low < high )**

**{**

**mid = low +(high - low)/2;**

**if(arr[mid]>=arr[mid+1])**

**{**

**high=mid;**

**}**

**else**

**{**

**low=mid+1;**

**}**

**}**

**return low;**

**}**

**int main()**

**{**

**int mountainArray[]= {4,8,16,32,27,9,3};**

**int n = sizeof(mountainArray)/sizeof(mountainArray[0]);**

**int peak=peakIndex(mountainArray,n);**

**cout<<"Peak index is:"<<peak;**

**return 0;**

**}**

25. Search insert position

**Solution:-**

#include <stdio.h>

int find\_index(int arr[], int n, int K)

{

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

if (arr[i] == K)

return i; else if (arr[i] > K)

return i;

return n;

}

int main()

{

int arr[] = { 1, 3, 5, 6 };

int n = sizeof(arr) / sizeof(arr[0]);

int K = 2;

printf("%d\n", find\_index(arr, n, K));

return 0;

}

**26. Robot return to origin**

Solution:

#include <stdio.h>

public class Main {

public static void main(String[] args) {

Main main = new Main();

boolean result = main.judgeCircle("RLUD");

System.out.println(result);

}

public boolean judgeCircle(String moves) {

char[] array = moves.toCharArray();

int U = 0;

int R = 0;

for(int i=0; i<array.length; i++){

if(array[i] == 'U'){

U++;

}else if(array[i] == 'D'){

U--;

}else if(array[i] == 'R'){

R++;

}else if(array[i] == 'L'){

R--;

}

}

if(U == 0 && R == 0){

return true;

}

return false;

}

}

return 0;

}

**27. Longest common prefix**

Solution:-

#include <stdio.h>

void findprefix(char \*str1, char \*str2, char \*found);

int main(void) {

char str1[100];

char str2[100];

char found[10] = { '\0' };

printf("\nGive string 1: ");

scanf("%99s", str1);

printf("\nGive string 2: ");

scanf("%99s", str2);

findprefix(str1, str2, found);

printf("%s", found);

return 0;

}

void findprefix(char \*str1, char \*str2, char \*found) {

int i, j;

for () {

if () {

}

}

}

**28. Number of string that appears as a substring program**

**Solution:-**

#include <stdio.h>

#include <string.h>

char str[100], sub[100];

int count = 0, count1 = 0;

void main()

{

int i, j, l, l1, l2;

printf("\nEnter a string : ");

scanf("%[^\n]s", str);

l1 = strlen(str);

printf("\nEnter a substring : ");

scanf(" %[^\n]s", sub);

l2 = strlen(sub);

for (i = 0; i < l1;)

{

j = 0;

count = 0;

while ((str[i] == sub[j]))

{

count++;

i++;

j++;

}

if (count == l2)

{

count1++;

count = 0;

}

else

i++;

}

printf("%s occurs %d times in %s", sub, count1, str);

}

**29. Check if string halves are alike program**

**Solution:-**

#include <stdio.h>

#include <string.h>

string vowels = "aeiouAEIOU";

class Solution {

public:

bool halvesAreAlike(string S) {

int mid = S.size() / 2, ans = 0;

for (int i = 0, j = mid; i < mid; i++, j++) {

if (~vowels.find(S[i])) ans++;

if (~vowels.find(S[j])) ans--;

}

return ans == 0;

}

};

}

**30. Largest sum contiguous subarray program**

**Solution:-**

#include <stdio.h>

int maxSubArraySum(int a[], int size)

{

int current\_sum = 0, maximum\_sum = 0;

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

{

current\_sum = current\_sum + a[i];

if (current\_sum > maximum\_sum)

maximum\_sum = current\_sum;

if (current\_sum< 0)

current\_sum = 0;

}

return maximum\_sum;

}

int main()

{

int a[] = {-2, -3, 4, -1, -2, 1, 5, -3};

int n = sizeof(a)/sizeof(a[0]);

int max\_sum = maxSubArraySum(a, n);

printf("Maximum sum of the contiguous array is : %d", max\_sum);

return 0;

}

**31. Maximum length sequence of continuous ones program**

**Solution:-**

#include <stdio.h>

int max(int x, int y) {

return (x > y) ? x : y;

}

int findIndexofZero(int arr[], int n)

{

if (n == 1) {

return (arr[0] == 0 ? 0 : -1);

}

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

{

if (arr[i] == 1) {

arr[i] += arr[i - 1];

}

}

int count = 0;

for (int i = n - 1; i >= 0; i--)

{

count = max(arr[i], count);

if (arr[i]) {

arr[i] = count;

}

else { count = 0;

}

}

int max\_count = 0;

int max\_index = -1;

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

{

if (arr[i] == 0)

{if (i == 0)

{

if (max\_count < arr[i + 1] + 1)

{

max\_count = arr[i + 1] + 1;

max\_index = i;

}

}

else if (i == n - 1)

{

if (max\_count < arr[i - 1] + 1)

{

max\_count = arr[i - 1] + 1;

max\_index = i;

}

}

else if (max\_count < arr[i - 1] + arr[i + 1] + 1)

{

max\_count = arr[i - 1] + arr[i + 1] + 1,

max\_index = i;

}

}

}

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

{

if (arr[i]) {

arr[i] = 1;

}

}

return max\_index;

}

int main(void)

{

int arr[] = { 0, 0, 1, 0, 1, 1, 1, 0, 1, 1 };

int n = sizeof(arr) / sizeof(arr[0]);

int index = findIndexofZero(arr, n);

if (index != -1) {

printf("Index to be replaced is %d", index);

}

else {

printf("Invalid input");

}

return 0;

}

**32.Factorial Number**

**Solution**:-

#include<stdio.h>

long factorial(int n)

{

if (n == 0)

return 1;

else

return(n \* factorial(n-1));

}

void main()

{

int number;

long fact;

printf("Enter a number: ");

scanf("%d", &number);

fact = factorial(number);

printf("Factorial of %d is %ld\n", number, fact);

return 0;

}

**33. Friends pairing program**

**Solution:-**

#include <iostream>

using namespace std;

int countPairs(int n) {

int pairs[n + 1];

pairs[0] = 0;

pairs[1] = 1;

pairs[2] = 2;

for (int i = 3; i <= n; i++)

pairs[i] = pairs[i-1] + (i-1) \* pairs[i-2];

return pairs[n];

}

int main() {

int n;

cout << "Enter numbers: "; cin >> n;

cout << "Number of ways to pair "<<n<<" friends: " << countPairs(n);

}