PROGRAM NO: 1

AIM: Simple program using formatted I/O.

ALGORITHM

Step 1 : start

Step 2 : read values of a and b from user

Step 3 : print the values of a and b

Step 4 : read value of n from user

Step 5 : print the value of n, right justified in the field

of 6 columns

Step 6 : print the value of n, right justified in the field

of 2 columns

Step 7 : print the value of n, left justified in the field

of 6 columns

Step 8 : print the value of n, placing zeros before n in the field

of 6 columns

Step 9 : read values of x and y from user

Step 10 : read value of w from user

Step 11 : read value of p from user

Step 12 : print the values of x, is rounded to p decimal places

and right justified in the field of w columns

Step 13 : print the value of x in exponential form with

default precision 6

Step 14 : print the value of y, is rounded to 2 decimal places and

left justified in the field of 7 columns

Step 15 : print the value of y in exponential form , is rounded

to 2 decimal places and right justified in the field

of 10 columns

Step 16 : read value of str from user

Step 17 : print the value of str in the field of 20 columns 3

Step 18 : print the first 10 characters of the string str in the field

of 30 columns and right justified

Step 19 : print the first 5 characters of the string str

Step 20 : print the first 10 characters of the string str in the field

of 20 columns and left justified

Step 21 : print the value of str

Step 22 : stop

PROGRAM

#include<stdio.h>

void main()

{

int a,b,d,w,p;

float x;

double y;

char str[20];

printf("FORMATTED INPUT & OUTPUT\n");

printf("Enter three integer numbers\n");

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

printf("value in a is%d\n Value in b is %d\n",a,b);

printf("Enter a 4 digit number\n");

scanf("%4d",&d);

printf("%6d\n",d);

printf("%2d\n",d);

printf("%-6d\n",d);

printf("%06d\n",d);

printf("Enter two floating point number\n");

scanf("%f%f",&x,&y);

printf("Enter the field width\n");

scanf("%d",&w);

printf("Enter the precision\n");

scanf("%d",&p);

printf("%\*.\*f\n",w,p,x);

printf("%e\n",x);

printf("%-7.2f\n",y);

printf("%10.2e\n",y);

printf("Enter a string\n");

scanf("%s",str);

printf("%20s\n",str);

printf("%30.10s\n",str);

printf("%.5s\n",str);

printf("%-20s\n",str);

printf("%5s\n",str);

}

OUTPUT

FORMATTED INPUT & OUTPUT

Enter three integer numbers

3

9

8

value in a is3

Value in b is 8

Enter a 4 digit number

2

2

2

2

000002

Enter two floating point number

7

6

Enter the field width

2

Enter the precision

1

7.0

7.000000e+00

0.00

5.37e-315

Enter a string

cat

cat

cat

cat

cat

cat

PROGRAM NO :2

AIM : Write a program to reverse a number using typedef

ALGORITHM

Step1 : start

Step 2: declare int number as typedef

Step 3:declare n, temp, revrse=0 , r as number

Step 4:display enter a number to find reverse

Step 5: read n

Step 6:store n in temp

Step 7 : when temp > 0 goto step 8,9,10

Step 8: r=temp%10;

Step 9: revrse=(revrse\*10)+r;

Step 10: temp=temp/10;

Step 11: print reverse number

Step 12: stop

PROGRAM

#include<stdio.h>

main()

{

typedef int number;

number n,temp,revrse=0,r;

printf("\n Program to find reverse number");

printf("\nPlease enter any Number: ");

scanf("%d",&n);

temp=n;

while(temp>0)

{

r=temp%10;

revrse=(revrse\*10)+r;

temp=temp/10;

}

printf("\nreverse of %d is= %d",n,revrse);

return 0;

}

OUTPUT

Program to find reverse number

Please enter any Number: 675

reverse of 675 is= 576

PROGRAM NO: 3

AIM: Find grade of a student using else if ladder.

ALGORITHM

Step 1 : start

Step 2 : read values of m1,m2and m3 from user

Step 3 : Find average by add m1,m2 and m3 then divide by 3

Step 4: if average is greater than 90 then print A+ else go to step 5

Step 5 : if average greater than or equal to 80 and less than 90 print B+ else go to Step 6

Step 6 : if average is greater than or equal to 70 and less than 80 print C+ else go to step 7

Step 7 : if average is greater than or equal to 60 and less than 70 print D+ else go to step 8

Step 8 : Print „Fail „ and go to step 9

Step 9 : stop

PROGRAM

#include<stdio.h>

void main()

{

int m1,m2,m3,avg;

printf("Enter three marks\n");

scanf("%d%d%d",&m1,&m2,&m3);

avg=(m1+m2+m3)/3;

if(avg>=90)

printf("A+\n");

else if(avg>=80&&avg<90)

printf("B+\n");

else if(avg>=70&&avg<80)

printf("C+\n");

else if(avg>=60&&avg<70)

printf("D+\n");

else

printf("fail\n");

}

OUTPUT

Enter three marks

50

87

90

C+

PROGRAM NO :4

AIM : Program to print numbers equivalent to Roman Numbers using Switch case

ALGORITHM

Step 1 : start

Step 2 : declare ch as character

Step 3 : display enter the Roman Character

Step 4 : read ch

Step 5: select choice ch

Step 6 : if ch equal to I display 1

Step 7 : if ch equal to V display 5

Step 8 : if ch equal to X display 10

Step 9 : if ch equal to L display 50

Step 10 : if ch equal to C display 100

Step 11 : if ch equal to D display 500

Step 12 : if ch equal to F display 100

Step 13 : print “Invalid entry”

Step 14: stop

PROGRAM

#include<stdio.h>

main()

{

char ch;

printf("Enter Roman number \n");

scanf("%c",&ch);

switch(ch)

{

case 'I':

printf("1");

break;

case 'V':

printf("5");

break;

case 'X':

printf("10");

break;

case 'L':

printf("50");

break;

case 'C':

printf("100");

break;

case 'D':

printf("500");

break;

case 'F':

printf("1000");

break;

default:

printf("invalid entry");

}

}

OUTPUT

Enter Roman number

V

5

PROGRAM NO: 5

AIM: Find Armstrong numbers within a range using while loop.

ALGORITHM

Step 1 : start

Step 2 : read value of l from user

Step 3 : set i=l

Step 4 : Repeat while number less than or equal to l

Step 5 : set n=I and s=0

Step 6 : Repeat while n>0

Step 7 : calculate a=n%10

Step 8 : calculate p=d\*d\*d ,s=s+p and n=n/10.

Step 9 : check s==1 if it is true go to step 10

Step 10 : print value of i

Step 11 : increment I by 1 and go to step 4

Step 12 : stop

PROGRAM

#include<stdio.h>

void main()

{

int l,n,d,i;

printf("Enter a limit:\n");

scanf("%d",&l);

printf("Armstrong numbers are:\n");

i=1;

while(i<=l)

{

int p,s;

n=i;

s=0;

while(n>0)

{

d=n%10;

p=d\*d\*d;

s=s+p;

n=n/10;

}

if(s==i)

{

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

}

i++;

}

}

OUTPUT

Enter a limit:

200

Armstrong numbers are:

1

153

PROGRAM NO: 6

AIM : program to find the sum of first n natural numbers using do while loop

ALGORITHM

Step1 :Start

Step 2: display enter the number of natural numbers

Step 3: Read value of n

Step 4: set i=0

Step 5:print value of I and sum= sum+ i

Step 6: increment value of I by 1

Step 7 : Repeat step 5,6 until i < 1

Step 8 : print value of sum

Step 9 : Stop

PROGRAM

#include<stdio.h>

void main()

{

int n,sum=0,i=1;

printf("enter the limit");

scanf("%d",&n);

do

{

sum=sum+i;

i++;

}

while(i<=n);

printf("sum of first n natural number is %d",sum);

}

OUTPUT

enter the limit5

sum of first n natural number is 15

PROGRAM NO: 7

AIM: Average of n numbers using goto statement.

ALGORITHM

Step 1 : start

Step 2 : read the value of n from user

Step 3 : set i equal to 0

Step 4 : if i greater than n then go to step 5 else go to step 9

Step 5 : read the value of p from user

Step 6 : if p less than 0 go to step 9 else go to step 7

Step 7 : calculate s equal to s+p

Step 8 : i equal to i + 1 go to step 4

Step 9 : calculate avg equal to (s/(i))

Step 10 : print the value of avg

Step 11 : stop

PROGRAM

#include<stdio.h>

main()

{

int n,p,i,s=0;

float avg;

printf("Enter limit\n");

scanf("%d",&n);

printf("Enter the elements\n");

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

{

scanf("%d",&p);

if(p<0)

{

goto label;

}

s=s+p;

}

label:

avg=(s/(i));

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

}

OUTPUT:

Enter limit 5

Enter the elements 5

4

8

10

5

Average=6.00000

PROGRAM NO: 8

AIM: Linear search using function.

ALGORITHM

Step 1: start

Step 2: read the value of n from user

Step 3: set i equal to 0

Step 4: if i less than n then go to step 5 else go to step 7

Step 5: read the value of a[i] from user

Step 6: i equal to i+1 got to step 4

Step 7: call function search ()

Step 8: if f equal to 0 print element not found

Step 9: Stop

Function search()

Step 1: start

Step 2: read the value of s from user

Step 3: set i equal to 0

Step 4: if i less than n then go to step 5 else go to step 8

Step 5: if a[i] equal to s then go to step 6 else go to step 7

Step 6: print the value of i+1 go to step 10

Step 7: Set flag value equal to 1

Step 8: i equal to i+1 got to step 4

Step 9: Return value of f

Step 10: stop

PROGRAM

#include<stdio.h>

int search();

int f=0,n,a[25],p,i,l;

void main()

{

printf("Enter the limit\n"); 34

scanf("%d",&n);

printf("enter the numbers\n");

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

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

search();

if(f==0)

printf("not found\n");

}

int search()

{ int s;

printf("enter the number to be searched\n");

scanf("%d",&s);

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

{

if(a[i]==s)

{

printf("number found at %d position\n",i+1);

f=1;

}

}

return f;

}

OUTPUT

Enter the limit

10

enter the numbers

1

3

4

6

7

8

4

3

2

5

enter the number to be searched

8

number found at 6 position

Enter the limit

5

enter the numbers

1

3

4

2

4

enter the number to be searched

5

not found

Enter the limit

5

enter the numbers

1

2

2

3

3

enter the number to be searched

2

number found at 2 position

number found at 3 position

PROGRAM:9

AIM: Selection sort using function.

ALGORITHM

Step 1: start

Step 2: read the value of n from user

Step 3: set i equal to 0

Step 4: if i less than n then go to step 5 else go to step 7

Step 5: read the value of a[i] from user

Step 6: i equal to i+1 got to step 4

Step 7: call function selection (a,n)

Step 8: stop

Function selection(a[ ],n)

Step 1: start

Step 2: set i equal to 0

Step 3: if i less than n then go to step 3 else go to step 12

Step 4: set j equal to i+1

Step 5: if j less than n-1 then go to step 6 else go to step 11

Step 6: if ( a[i] > a[j] ) go to step 7 else got to step 10

Step 7: set temp equal to a[i]

Step 8: set a[i] equal to a[j]

Step 9: set a[j] equal to temp

Step 10: j equal to j+1 go to step 5

Step 11: i equal to i+1 go to step 3

Step 12: set i equal to 0

Step 13: if i less than n then go to step 14 else go to step 16

Step 14: print the value of a[i]

Step 15: i equal to i+1 got to step 13

Step 16: stop

#include<stdio.h>

void selection(int [],int);

int i,j;

void main()

{

int a[10],n;

printf("Enter limit\n");

scanf("%d",&n);

printf("Enter numbers\n");

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

{

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

}

selection(a,n);

}

void selection(int a[],int n)

{

int temp;

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

{

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

{

if(a[i]>a[j])

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

}

}

}

printf("sorted numbers are\n");

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

{

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

}

}

OUTPUT

Enter limit

10

Enter numbers

34

23

1

2

65

45

29

78

44

8

sorted numbers are

1

2

8

23

29

34

44

45

65

78

PROGRAM NO:10

AIM: Bubble sort using function.

ALGORITHM

Step 1: start

Step 2: read the value of n from user

Step 3: set i equal to 0

Step 4: if i less than n then go to step 5 else go to step 7

Step 5: read the value of a[i] from user

Step 6: i equal to i+1 got to step 4

Step 7: call function bubble(a,n)

Step 8: stop

Function bubble(a[ ],n)

Step 1 : start

Step 2 : set i equal to 0

Step 3 : if i less than n then go to step 3 else go to step 12

Step 4 : set j equal to 0

Step 5 : if j less than n-i-1then go to step 6 else go to step 11

Step 6 : if ( a[j] > a[j+1] ) go to step 7 else got to step 10

Step 7 : set temp equal to a[j]

Step 8 : set a[j] equal to a[j+1]

Step 9 : set a[j+1] equal to temp

Step 10: j equal to j+1 go to step 5

Step 11: i equal to i+1 go to step 3

Step 12: set i equal to 0

Step 13: if i less than n then go to step 14 else go to step 16

Step 14: print the value of a[i]

Step 15: i equal to i+1 got to step 13

Step 16: stop

PROGRAM

#include<stdio.h>

void bubble(int [],int);

int i,j;

void main()

{

int a[10],n;

printf("Enter limit\n");

scanf("%d",&n);

printf("Enter numbers\n");

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

{

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

}

bubble(a,n);

}

void bubble(int a[],int n )

{

int temp;

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

{

for(j=0;j<(n-i-1);j++)

{

if(a[j]>a[j+1])

{

temp=a[j];

a[j]=a[j+1];

a[j+1]=temp;

}

}

}

printf("sorted numbers are\n");

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

{

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

}

}

OUTPUT

Enter limit

5

Enter numbers

3

7

9

7

4

sorted numbers are

3

4

7

7

9

PROGRAM:11:

AIM: sum of elements in an array

ALGORITHM

Step 1: start

Step 2: read the value of n from user

Step 3: set i equal to 0

Step 4: if i less than n then go to step 5 else go to step 7

Step 5: read the value of a[i] from user

Step 6: i equal to i+1 go to step 4

Step 7: set s equal to call function fun()

Step 8: print the value of s

Step 9: stop

Function fun()

Step 1: start

Step 2: set i equal to 0 and sum=0

Step 3: if i less than n then go to step 4 else go to step 6

Step 4: calculate sum equal to sum+a[i]

Step 5: i equal to i+1 go to step 3

Step 6: return sum

Step 7: stop

Program:

#include<stdio.h>

float fun();

int a[25],i,n;

void main()

{

int s;

printf("Enter a limit");

scanf("%d",&n);

printf("Enter numbers\n");

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

{

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

}

s=fun();

printf("sum=%d",s);

}

float fun()

{

int sum=0;

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

{

sum=sum+a[i];

}

return sum;

}

OUTPUT

Enter a limit

5

Enter numbers

4

3

2

3

2

sum=14

PROGRAM12:

AIM:implementing simple pointer

ALGORITHM

step 1: start

step 2: declare interger variable i equal to 0 and pointer variable \*p

step 3: declare floating point variable f equal to 3.4 and pointer variable

\*fp

step 4: declare char variable c equal to 'a' and a pointer \*cp

step 5: print i,f,c

step 6: ip equal to &f

step 7: print value of ip and \*ip

step 8: fp equal to &f

step 9: print values of fp and \*fp

step 10: cp equal to &c

step 11: print values of cp and \*cp

step 12: stop

PROGRAM

#include<stdio.h>

main()

{

int i=55,\*ip;

float f=45.77,\*fp;

char c='v',\*cp;

printf("i=%d\n",i);

printf("f=%f\n",f);

printf("c=%c\n",c);

ip=&i;

printf("values of i=%u\n",ip);

printf("values of i=%d\n",\*ip);

fp=&f;

printf("address of f=%u\n",fp);

printf("values of f=%f\n",\*fp);

cp=&c;

printf("address of c=%u\n",cp);

printf("values of c=%c\n",\*cp);

}

OUTPUT

i=55

f=45.770000

c=v

values of i=3215420288

values of i=55

address of f=3215420284

values of f=45.770000

address of c=3215420283

values of c=v

PROGRAM NO: 13

AIM: Matrix multiplication using function.

ALGORITHM

Step 1 : start

Step 2 : read the value of r1 and c1 from user

Step 3 : read the value of r2 and c2 from user

Step 4 : if r1 not equal to c2 then print multiplication not possible else go tostep 5

Step 5 : set i equal to 0

Step 6 : if i less than r2 then go to step 7 else go to step 12

Step 7 : set j equal to 0

Step 8 : if j less than c2 then go to step 9 else go to step 11

Step 9 : read the value of a[i][j] from user

Step 10: j equal to j+1 got to step 8

Step 11: i equal to i+1 go to step 6

Step 12: set i equal to 0

Step 13: if i less than r1 then go to step 14 else go to step 19

Step 14: set j equal to 0

Step 15: if j less than c2 then go to step 16 else go to step 18

Step 16: read the value of b[i][j] from user

Step 17: j equal to j+1 got to step 15

Step 18: i equal to i+1 go to step 13

Step 19: Print two metrices a[i][j] and b[i][j]

Step 20: Call function mul(a,b,r1,c2)

Step 21: stop

Function mul()

Step 1 : start

Step 2 : set i equal to 0

Step 3 : if i less than r1 then go to step 4 else go to step 13

Step 4 : set j equal to 0

Step 5 : if j less than c2 then go to step 6 else go to step 8

Step 6 : set m[i][j] equal to 0

Step 7 : set k equal to 0

Step 8 : if k less than r1 then go to step 9 else go to step 11

Step 9 : calculate m[i][j] equal to m[i][j]+(a[i][k]\*b[k][j])

Step 10: k equal to k+1 go to step 8

Step 11: j equal to j+1 go to step 5

Step 12: i equal to i+1 go to step 3

Step 13: set i equal to 0

Step 14: if i less than m then go to step 15 else go to step 20

Step 15: set j equal to 0

Step 16: if j less than n then go to step 17 else go to step 19

Step 17: print the value of m[i][j]

Step 18: j equal to j+1 go to step 16

Step 19: i equal to i+1 go to step 14

Step 20: stop

PROGRAM

#include<stdio.h>

void mul(int [][10],int [][10],int ,int);

inti,j,r1,c1,r2,c2,k;

void main()

{

int a[10][10],b[10][10];

printf("Enter row and column of first matrix\n");

scanf("%d%d",&r1,&c1);

printf("Enter row and column of second matrix\n");

scanf("%d%d",&r2,&c2);

if(r1!=c2)

printf("not possible\n");

else

{

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

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

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

}

}

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

for(i=0;i<r2;i++)

{

for(j=0;j<c2;j++)

{

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

}

}

printf("FIRST MATRIX\n");

for(i=0;i<r1;i++)

{

for(j=0;j<c1;j++)

{

printf("%d\t",a[i][j]);

}

printf("\n");

}

printf("SECOND MATRIX\n");

for(i=0;i<r2;i++)

{

for(j=0;j<c2;j++)

{

printf("%d\t",b[i][j]);

}

printf("\n");

}

mul(a,b,r1,c2);

}

}

void mul(int a[][10],int b[][10],int r1,int c2)

{

int m[10][10];

printf("multiplication table\n");

for(i=0;i<r1;i++)

{

for(j=0;j<c2;j++)

{

45 m[i][j]=0;

for(k=0;k<r1;k++)

{

m[i][j]=m[i][j]+(a[i][k]\*b[k][j]);

}

printf("%d\t",m[i][j]);

}

printf("\n");

}

}

OUTPUT

Enter row and column of first matrix

2

2

Enter row and column of second matrix

2

2

enter the elements of first matrix

2

1

2

1

enter the elements of second matrix

2

1

2

1

FIRST MATRIX

2 1

2 1

SECOND MATRIX

2 1

2 1

multiplication table

6 3

6 3

Enter row and column of first matrix

2

2

Enter row and column of second matrix

2

1

not possible

PROGRAM NO: 14

AIM: String manipulations.

ALGORITHM

Step 1 : start

Step 2 : read the value of s from user

Step 3 : find the length of string s then go to step 4

Step 4 : print the string length

Step 5 : read value of s1 and s2 from user

Step 6 : copy the string s2 to s1

Step 7 :print the value of s1

Step 8 : read value of s3 and s4from user

Step 9 : compare s3 and s4 if it is equal to 0 then go to step 10 else go to step 11

Step 10 : print „strings are equal‟

Step 11 : print „strings are not equal

Step 12 : concatenate strings s3 and s4

Step 13 : print value of s3

Step 14: stop

PROGRAM

#include<stdio.h>

#include<string.h>

void main()

{

char s[20],s1[10],s2[10],s3[10],s4[10];

int n,h;

printf("Enter the string:\n");

scanf("%s",s);

n=strlen(s);

printf("\*\*\*\*LENGTH\*\*\*\*\n");

printf("Length of string using strlen: %d\n",n);

printf("Enter two strings:\n");

scanf("%s%s",s1,s2);

strcpy(s1,s2);

printf("\*\*\*\*\*\*COPY\*\*\*\*\*\n");

printf("Copied string is:%s\n",s1);

printf("Enter two strings:\n");

scanf("%s%s",s3,s4);

printf("\*\*\*\*\*COMPARE\*\*\*\*\n");

h=strcmp(s3,s4);

if(h==0)

{

printf("Two strings are equal\n");

}

else

{

printf("Two strings are not equal\n");

}

strcat(s3,s4);

printf("\*\*\*\*\*CONCATENATION\*\*\*\*\n");

printf("The concatenated string:%s\n",s3);

}

OUTPUT:

Enter the string:

cat

\*\*\*\*LENGTH\*\*\*\*

Length of string using strlen: 3

Enter two strings:

cat

mat

\*\*\*\*\*\*COPY\*\*\*\*\*

Copied string is:mat

Enter two strings:

cat

mat

\*\*\*\*\*COMPARE\*\*\*\*

Two strings are not equal

\*\*\*\*\*CONCATENATION\*\*\*\*

The concatenated string:catmat

PROGRAM NO :15

Aim : program to find trace of a matrix

ALGORITHM

Step 1: start

Step 2:declare a[10][10],I,j,sum=0,m,n as integers

Step 3: display enter the values of m and n

Step 4:read m and n

Step 5:display enter the elements of the matrix

Step 6:for i=0 to m and j=0 to n read the elements of the matrix

Step 7: for i=0 to m and j=0 to n goto step 8

Step 8: if i = j add array elements to sum

Step 9: display sum

Step 10: stop

PROGRAM

#include<stdio.h>

main()

{

int a[10][10],i,j,sum=0,m,n;

printf("enter the values of m,n\n");

scanf("%d%d",&m,&n);

printf("enter the elements of matrix a\n");

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

{

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

{

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

}

}

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

{

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

{

if(i==j)

{

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

}

}

}

printf("Trace of a Matrix= %d ",sum);

}

OUTPUT

enter the values of m,n

2

2

enter the elements of matrix a

2

5

8

1

Trace of a Matrix= 3

PROGRAM:16

AIM:array of pointers

ALGORITHM

p 1 : start

Step 2 : read the value of n from user

Step 3 : set i equal to 0

Step 4 : if i less than n then go to step 5 else go to step 10

Step 5 : read the value of s[i] from user

Step 6 : i equal to i+1 got to step 4

Step 7 : set i equal to 0

Step 8 : if i less than n then go to step 9 else go to step 16

Step 9 : set j equal to i+1

Step 10: if j less than n then go to step 11 else go to step 15

Step 11: compare &s[i ]and &s[j] if it is greater than 0 then go to step 12 else go to step15

Step 12: set temp equal to s[i]

Step 13: set s[i] equal to s[j]

Step 14: set s[j] equal to temp

Step 15: j equal to j+1 go to step 10

Step 16: i equal to i+1 go to step 8

Step 17: print the value of s[i]

Step 18: stop

PROGRAM

#include<stdio.h>

void main()

{

char \*s[20],\*temp;

int i,j,n;

printf("Enter limits:\n");

scanf("%d",&n);

printf("Enter names:\n");

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

{

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

}

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

{

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

{

if(strcmp(&s[i],&s[j])>0)

{

temp=s[i];

s[i]=s[j];

s[j]=temp;

}

}

}

printf("Sorted names are:\n");

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

{

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

}

}

OUTPUT

Enter limits:

5

Enter names:

kichu

appu

ammu

devu

anni

Sorted names are:

ammu

anni

appu

devu

kichu

PROGRAM 17:

AIM:pointer to function

ALGORITHM

step 1: start

step 2: declare integer a,b,s,(\*fnp)(int,int)

step 3: read value a and b from user

step 4: print a and b

step 5: set fnp eqal to sum

step 6: set s equal to (\*fnp)(a,b)

step 7: print value s

step 8: stop

Function sum()

step 1: start

step 2: declare integer s

step 3: set s equal to a+b

step 4: return s

step 5: stop

PROGRAM

include<stdio.h>

int sum(int,int);

main()

{

int a,b,s,(\*fnp)(int,int);

printf("enter two numbers");

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

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

fnp=sum;

s=(\*fnp)(a,b);

printf("sum=%d",s);

}

int sum(int a,int b)

{

int s;

s=a+b;

return s;

}

OUTPUT

enter two numbers

12

3

a=12 b=3sum=15

PROGRAM NO: 18

AIM: Implementation of pointer to pointer

ALGORITHM

Step 1 : start

Step 2 : set p1 equal to &a

Step 3 : set p2 equal to &p1

Step 4 : set q1 equal to &b

Step 5 : set q2 equal to &q1

Step 7 : read the value of a and b

Step 8 : calculate c equal to \*\*p2 + \*\*q2

Step 9 : print the value of c

Step 10 : calculate c equal to \*\*p2 - \*\*q2

Step 11 : print the value of c

Step 12 : calculate c equal to \*\*p2 \* \*\*q2

Step 13 : print the value of c

Step 14 : calculate c equal to \*\*p2 / \*\*q2

Step 15 : print the value of c

Step 16 : stop

PROGRAM

#include<stdio.h>

void main()

{

int a,b,c,\*p1,\*\*p2,\*q1,\*\*q2;

p1=&a;

p2=&p1;

q1=&b;

q2=&q1;

printf("Enter two numbers\n");

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

c=\*\*p2+\*\*q2; 57

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

c=\*\*p2-\*\*q2;

printf("Difference=%d\n",c);

c=\*\*p2\*\*\*q2;

printf("product=%d\n",c);

c=\*\*p2/ \*\*q2;

printf("division=%d\n",c);

}

OUTPUT

Enter two numbers

2

2

sum=4

Difference=0

product=4

division=1

PROGRAM NO:19

AIM:multiplication table

#include<stdio.h>

main()

{

int i,n;

printf("enter the number:");

scanf("%d",&n);

printf("\nMultiplication table of %d....",n);

for(i=1;i<=10;i++)

{

printf("\n\t%d \* %d = %d",i,n,i\*n);

}

}

OUTPUT

enter the number:

5

Multiplication table of 5....

1 \* 5 = 5

2 \* 5 = 10

3 \* 5 = 15

4 \* 5 = 20

5 \* 5 = 25

6 \* 5 = 30

7 \* 5 = 35

8 \* 5 = 40

9 \* 5 = 45

10 \* 5 = 50