***Bytes of Datatypes***

**Char-->1 bytes**

**Short int-->2 bytes**

**Long int -->4 bytes \*INT\***

**Long long int -->8 bytes**

**float-->4 bytes**

**double-->8 bytes**

**Long double-->16 bytes**

**1.SUM OF DIGITS RIGHT SIDE TO THE ELEMENT IN A ARRAY**

#include<stdio.h>

int main()

{

int N,arr[1000],ind,sum=0;

scanf("%d",&N);

for(ind=0;ind<=N-1;ind++)

{

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

sum=sum+arr[ind];

}

for(ind=0;ind<=N-1;ind++)

{

sum=sum-arr[ind];

printf("%d ",sum);

}

return 0;

}

**2.REMOVE AN ELEMENT FROM THE ARRAY**

#include<stdio.h>

int main()

{

int N,arr[1000],ind,val,i;

scanf("%d",&N);

for(ind=0;ind<=N-1;ind++)

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

scanf("%d",&val);

for(ind=0;ind<=N-1;ind++)

{

if(arr[ind]==val)

{

for(i=ind;i<N-1;i++)

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

N=N-1;

break;

//remove break for delete all val values

//with break for deleting one val values

}

}

for(ind=0;ind<=N-1;ind++)

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

return 0;

}

**3.PRINT STAR TRIANGLE**

include<stdio.h>

int main()

{

int n, s, x, y;

scanf("%d",&n);

for(x = 1; x <= n; x++)

{

for(s = x; s < n; s++)

printf(" ");

for(y = 1; y <= x; y++)

printf("\* ");

printf("\n");

}

return 0;

}

**4.SWAPPING 2 NUMBER:**

//one line format

#include<stdio.h>

int main()

{

int n, s, x, y;

n=10;

s=20;

n=(n+s)-(s=n);

printf("%d %d",n,s);

}

//normal format using 3rd variable

#include<stdio.h>

int main()

{

int n, s, x, y;

n=10;

s=20;

x=n; //

n=s;

s=x;

printf("%d %d",n,s);

}

//without using third var

#include<stdio.h>

int main()

{

int n, s, x, y;

n=10;

s=20;

n=n+s;

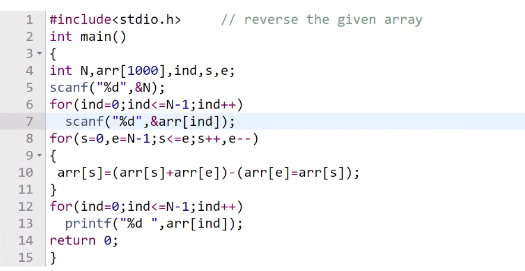
s=n-s;

n=n-s;

printf("%d %d",n,s);

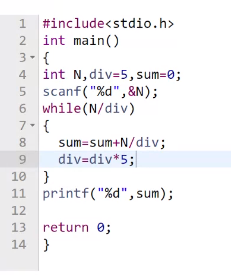
}

**5.REVERSE THE ARRAY**

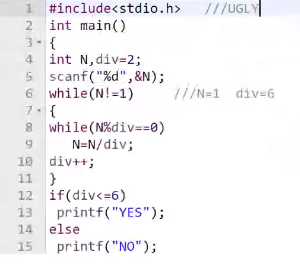


**6.TRAILING ZEROS IN THE FACTORIAL OF GIVEN NUMBER**

Input: 100 output: 24



**7.UGLY NUMBER**



**8.BETHORED NUMBER**

#include <stdio.h>

int main(void) {

int num1,num2,s1=0,s2=0,i,j;

scanf("%d",&num1);

scanf("%d",&num2);

for(i=1;i<num1;i++)

{

if(num1%i==0)

s1=s1+i;

}

for(j=1;j<num2;j++)

{

if(num2%j==0)

s2=s2+j;

}

if(s1==num2+1 && s2==num1+1)

printf("YES");

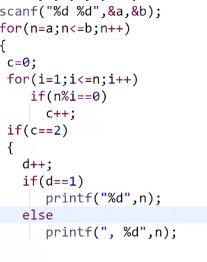
else

printf("NO");

return 0;

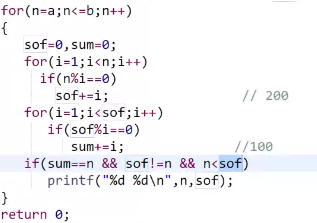
}

**9.PRIME NUMBER BETWEEN RANGE**



**10.AMBICABLE PAIR**

Input: 1 to 300 output:220 284



**11.BOX or SPIRAL Pattern**

4444444

4333334

4322234

4321234

4322234

4333334

4444444

#include <stdio.h>

int main(void) {

int N,row,col,spc;

scanf("%d",&N);

for(row=N;row>=1;row--) //decre

{

for(spc=N;spc>=row+1;spc--) //left

printf("%d",spc);

for(col=1;col<=2\*row-1;col++) //mid

printf("%d",row);

for(spc=row+1;spc<=N;spc++) //right

printf("%d",spc);

printf("\n");

}

for(row=2;row<=N;row++) //incre

{

for(spc=N;spc>=row+1;spc--) //left

printf("%d",spc);

for(col=1;col<=2\*row-1;col++) //mid

printf("%d",row);

for(spc=row+1;spc<=N;spc++) //right

printf("%d",spc);

printf("\n");

}

return 0;

}

**12.PATTERN**

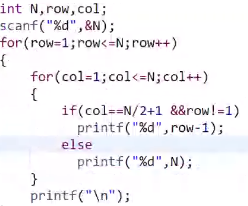
Input :3

o/p:

333

313

323



**13.PATTERN**

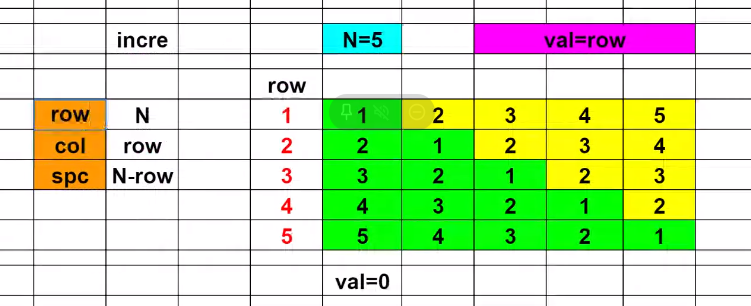
Input :3

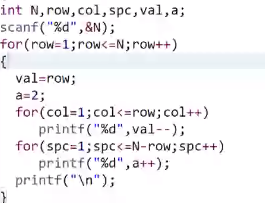
o/p:

123

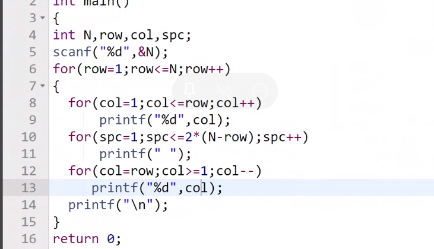
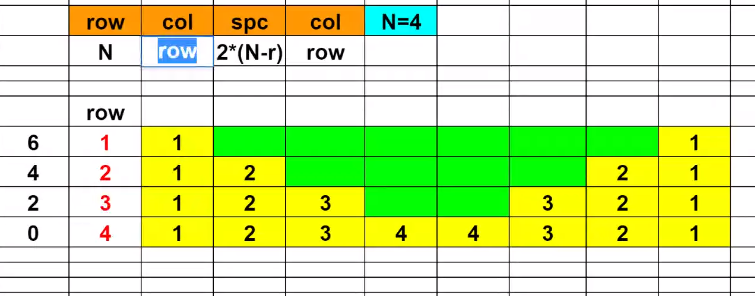
212

321





**14.PATTERN**

****

**15.PATTERN**

i/p:7 5

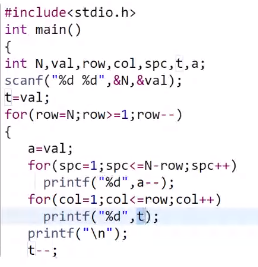
77777

76666

76555

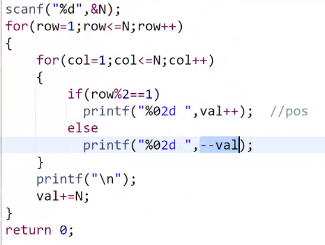
76544

76543

****

**16.ZIG ZAG PATTERN(IMPORTANT)**

****

****

**17.SANDWITCH PATTERN**

Input: 5

o/p:

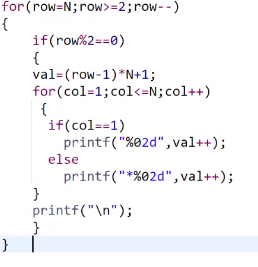
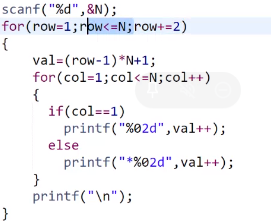
01\*02\*03\*04\*05

11\*12\*13\*14\*15

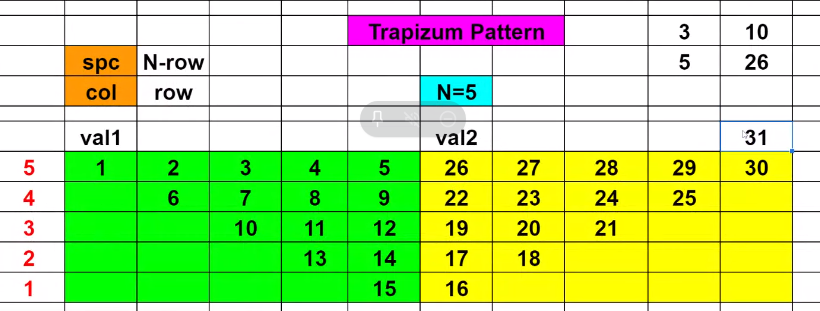
21\*22\*23\*24\*25

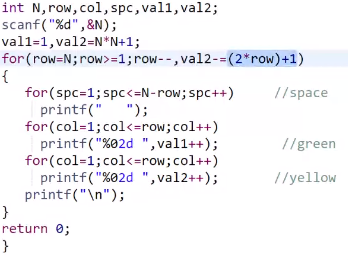
16\*17\*18\*19\*20

06\*07\*08\*09\*10

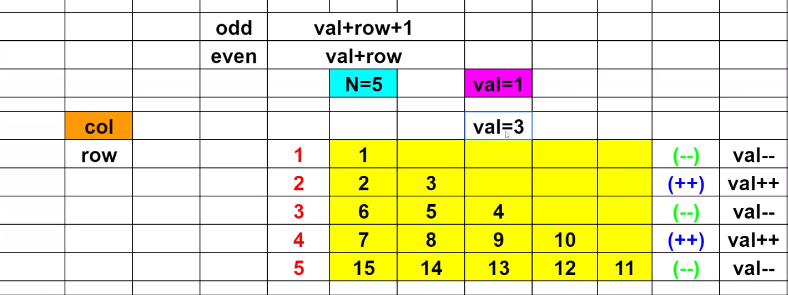


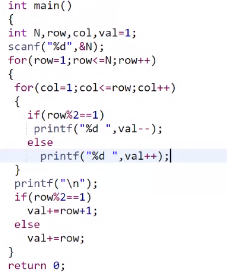
**18.TRAPEZIUM PATTERN**

****

****

**19.PATTERN**

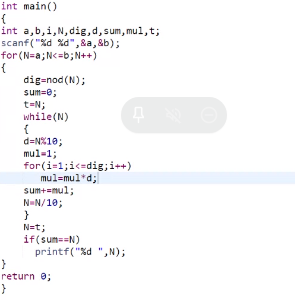
****

****

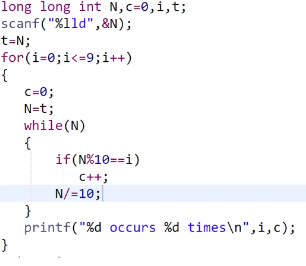
**20.ARMSTRONG NUMBER BETWEEN RANGE**

INPUT:1 200 OUTPUT:1 2 3 4 5 6 7 8 9 153

NOD IS TO BE FOUND USING ANOTHER FUNCTION

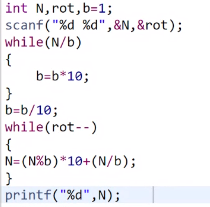
****

**21.FREQUENCY OF DIGITS IN NUMBER**

****

**22.LEFT ROTATION**

INPUT:12345 4 OUTPUT: 51234

****

**23.WRITE A PROGRAM TO CALCULATE ELAPSED TIME IN SECONDS**

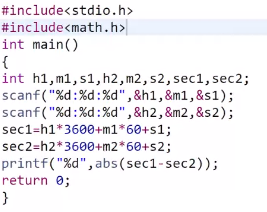
INPUT:(HH:MM:SS HH:MM:SS)

01:00:00

02:00:00

OUTPUT:(SECONDS)

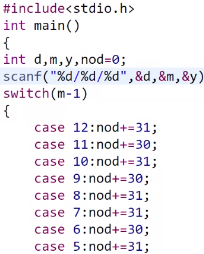
3600

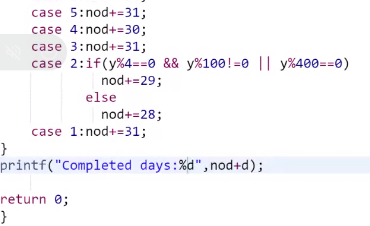


**24.COMPLETED DAYS**

INPUT:14/02/1999

OUTPUT:COMPLETED DAYS:45

****

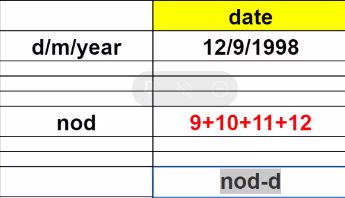
****

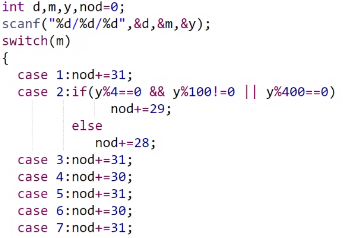
**25.REMAINING DAYS:**

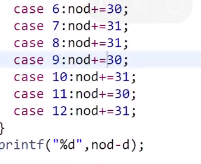
INPUT

01/01/1999

OUTPUT:364







**26.CALCULATE YEAR,MONTH,DAYS FOR GIVEN NUMBER**

Sample input:1000

Output:

year=2

week=38

days=4

#include <stdio.h>

int main(void) {

int N=1000;

int y,w,d;

y=N/365;

w=(N%365)/7;

d=(N%365)%7;

printf("Year: %d\nWeek: %d\nDays: %d",y,w,d);

return 0;

}

**27.CALCULATE NO.OF DAYS BETWEEN 2 DATES IN DIFFERENT YEAR**

Sample input:

4/2/2018

4/2/2020

Output:

No. of days between 2 dates of different year: 730

#include <stdio.h>

int main(void)

{

int d1,m1,y1,d2,m2,y2,nod=0;

scanf("%d/%d/%d",&d1,&m1,&y1);

scanf("%d/%d/%d",&d2,&m2,&y2);

//completed Year

for(int y=y1+1;y<=y2-1;y++)

{

if(y%4==0 && y%100!=0 || y%400==0)

{

nod+=366;

}

else

nod+=365;

}

//completed Month

switch(m1)

{

case 1:nod+=31;

case 2:if(y1%4==0 && y1%100!=0 || y1%400==0)

nod+=29;

else

nod+=28;

case 3:nod+=31;

case 4:nod+=30;

case 5:nod+=31;

case 6:nod+=30;

case 7:nod+=31;

case 8:nod+=31;

case 9:nod+=30;

case 10:nod+=31;

case 11:nod+=30;

case 12:nod+=31;

}

nod-=d1;

switch(m2-1)

{

case 12:nod+=31;

case 11:nod+=30;

case 10:nod+=31;

case 9:nod+=30;

case 8:nod+=31;

case 7:nod+=31;

case 6:nod+=30;

case 5:nod+=31;

case 4:nod+=30;

case 3:nod+=31;

case 2:if(y2%4==0 && y2%100!=0 || y2%400==0)

nod+=29;

else

nod+=28;

case 1:nod+=31;

}

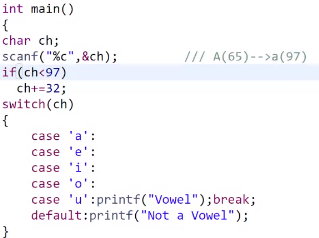
nod+=d2;

printf("No. of days between 2 dates of different year: %d",nod);

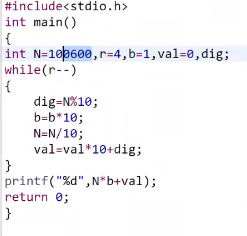
return 0;

}

**28.PRINT VOWEL OR NOT**



**29.REVERSE THE NUMBER BEtWEEN SPECIFIC VALUE**



**30.KAPREKAR NUMBER**

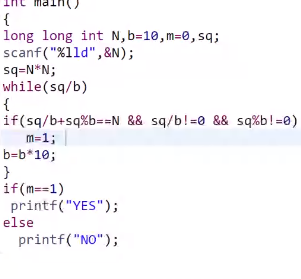
SAMPLE INPUT:

45

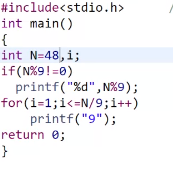
(45^2=2025 (DIVIDE INTO 2 PARTS)(ADD BOTH ) IF IT IS EQUAL TO N THEN PRINT YES)

OUTPUT:

YES



**31.TO PRINT SMALLEST NUMBER**



**32.MONOTONIC ARRAY**

(TO CHECK THE ARRAYS GOES IN FULLY INCREMENT OR DECREMENT NOT BOTH)

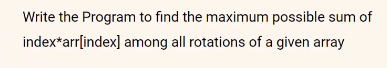
SAMPLE INPUT:

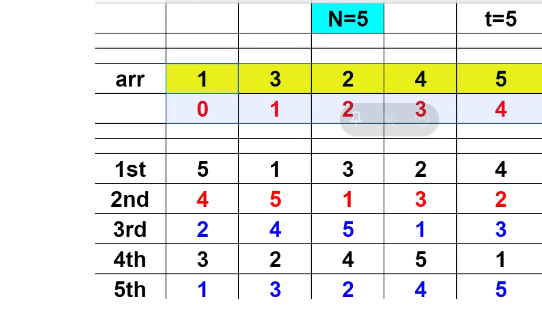
7

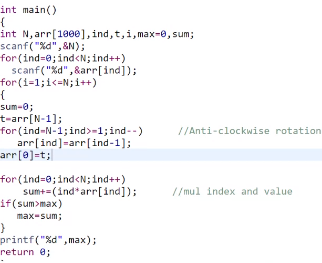
3 4 5 6 7 8 9 MONOTONIC



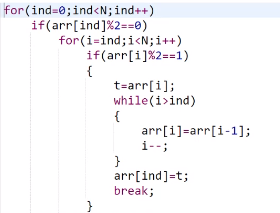
**33.MAXIMUM**







**34.ODD EVEN SEGREGATION**

****