

//Scanf and Print function for multiple digit  
;An assembly program to read 16 bit input

;and print that number again from and to  
consol

ORG 100H

MAIN PROC

START:

CALL SCAN

CALL PRINT

;Print a new line

MOV AH, 2

MOV DL, 10

INT 21H

MOV DL, 13

INT 21H

JMP START ;Take input again

RET

ENDP

;A procedure that reads a 16 bit signed input  
;and store that in AX

SCAN PROC

;Backup register values in stack

PUSH BX

PUSH CX

PUSH DX

;Clear register values

XOR BX, BX

XOR CX, CX

;Read first character

MOV AH, 1

INT 21H

;Check if it is a sign or digit

CMP AL, '-'

JE NEGATIVE

CMP AL, '+'

JE POSITIVE

JMP INPUT

NEGATIVE:

;Store that it is negative number in CX

MOV CX, 1

POSITIVE:

;Take a digit input if first input is sign

INT 21H

INPUT:

;Convert the digit ASCII to number

AND AX, 000FH

;As multiplication erases value in AX

```

;backup the digit to stack
PUSH AX

;Multiply previous value by 10 and add new
value
MOV AX, 10
MUL BX

;Pop new digit from stack
POP BX
ADD BX, AX

;Read digit repeatedly until space or carriage
return read
MOV AH, 1
INT 21H

CMP AL, ' '
JE ENDINPUT

CMP AL, 13
JE CARRIAGERETURN
JMP INPUT

CARRIAGERETURN:
;If last input is carriage return, print a new
line
MOV AH, 2
MOV DL, 10
INT 21H

;Store the positive input to AX
ENDINPUT:
MOV AX, BX

```

```

;Check if the value is negative
CMP CX, 0
JE ENDSCAN
NEG AX

ENDSCAN:
;Restore register values from stack
POP DX
POP CX
POP BX
RET
ENDP

; A procedure that prints number stored in AX in
decimal format
PRINT PROC
;Backup register values in stack
PUSH AX
PUSH BX
PUSH CX
PUSH DX

;Check if Ax is positive or negative
CMP AX, 0
JGE INIT

PUSH AX
MOV DL, '-'
MOV AH, 2

```

INT 21H	POP BX
POP AX	POP AX
NEG AX	RET
	ENDP

INIT:

XOR CX, CX ;Clear CX. Holds number of digits

MOV BX, 10 ;Holds divisor

DIGITIFY:

CWD ;Clear DX

DIV BX

PUSH DX ;Push last digit to stack

INC CX

;Check if the quotient is zero

CMP AX, 0

JNZ DIGITIFY

;Pop and print

MOV AH, 2

PRINTLOOP:

POP DX

OR DL, 30H ;Convert to ASCII

INT 21H

LOOP PRINTLOOP

;Restore register values from stack

POP DX

POP CX

##### Uva prblm solved by c and  
assembly#####

10035 Uva problem solved by using c:

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int a,b,r1,r2;
```

```
    while(1)
```

```
    {
```

```
int sum=0,c=0;
```

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

```
    if(a==0&&b==0)
```

```
        break;
```

```
    while(a>0&&b>0)
```

```
    {
```

```
        r1=a%10;
```

```
        r2=b%10;
```

```
        sum = r1+r2;
```

```
        a=a/2;
```

```
        b=b/2;
```

```
        if(sum+c>=10)
```

```
        {
```

```
            c++;
```

```
        }
```

```
    }
```

```
    if(c>0)
```

```
    {
```

```
        printf("%d carry operation\n",c);
```

```
    }
```

```
    else
```

```
        printf("No carry operation \n");
```

```
    }
```

```
    return 0;
```

```
}
```

10035 Uva problem solved by using assembly:

;UVA PRBLM 10035

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
SUM DW 0
```

```
A DW 0
```

```
B DW 0
```

```
R1 DW 0
```

```
R2 DW 0
```

```
C dw ?
```

```
D DW 0
```

```
E DW 0
```

```
P DW 0
```

```
.CODE
```

```
MAIN PROC
```

```
STRT:
MOV AX,@DATA ;DATA SEGMENT INITIALIZE
MOV DS,AX
```

```
INCLUDE 'EMU8086.INC'
```

```
XOR CX,CX
XOR AX,AX
XOR BX,BX ;O ASIGN ALL THE RESISTER
XOR DX,DX
```

```
MOV D,10
MOV E,2
```

```
CALL SCAN_NUM
;ADD P,CX
MOV A,CX
```

```
PRINTN
```

```
CALL SCAN_NUM
; ADD Q,CX
MOV B,CX
PRINTN
```

```
CMP A,0
JE EXIT
```

```
CMP B,0
JE EXIT
```

```
WHILE:
CMP A,0
JG CHECK
```

```
CHECK:
CMP B,0
JG CAL
JMP IF
```

```
CAL:
XOR AX,AX
ADD AX,A
DIV D
MOV R1,DX
XOR DX,DX ;R1
XOR AX,AX
```

```
ADD AX,B
DIV D
MOV R2,DX ;R2
XOR DX,DX
XOR AX,AX
```

```
MOV AX,R1 ;R2+R1
ADD AX,R2
MOV SUM,AX
```

XOR AX,AX

MOV AX,A

DIV E

MOV A,AX

XOR AX,AX ;A/2

XOR DX,DX

MOV AX,B

DIV E

MOV B,AX

XOR AX,AX

XOR DX,DX ;B/2

MOV AX,SUM

ADD AX,C

CMP AX,10

JGE L2

JMP WHILE

L2:

INC C

JMP WHILE

IF:

CMP C,0

JG PRINT1

PRINT1:

MOV AX,C

CALL PRINT\_NUM

PRINTN "CARRRY OPERATION"

JMP EXIT

PRINT2:

PRINTN " NO CARRY OPERATION"

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN

END MAIN

```
//UVA PRBLM 10079
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int p;
```

```
    long long int n;
```

```
    while(scanf("%lld",&n))
```

```
    {
```

```
        if(n<0)
```

```
            break;
```

```
        printf("%lld\n",1+ (n*(n+1)/2));
```

```
    }
```

```
    return 0;
```

```
}
```

```
;UVA PRBLM 10079
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
A DW 0
```

```
N DW 0
```

```
B DW ?
```

```
R DW ?
```

```
K DW ?
```

```
.CODE
```

```
MAIN PROC
```

```
STRT:
```

```
MOV AX,@DATA
```

```
MOV DS,AX ;DATA INITIALIZATION
```

```
INCLUDE 'EMU8086.INC'
```

```
;FOR USER INPUT
```

```
XOR CX,CX
```

```
XOR BX,BX ;CLEAR REGISTER
```

```
XOR AX,AX
```

```
XOR DX,DX
```

```
MOV BX,2
```

```
CALL SCAN_NUM ;SCAN N
```

```
PRINT
```

```
MOV AX,CX
```

```
ADD A,AX
```

```
CMP AX,0
```

```
JE EXIT ;N<0 BREAK
```

```
ADD AX,1 ;ELSE N+1
```

MUL A ;N*N+1	int v,t;
	while(scanf("%d %d",&v,&t)==2)
DIV BX ;/2	{
	if(v==0 && t==0)
ADD AX,1 ;+1	printf("0n");
	else
PRINTN " "	printf("%dn",2*v*t);
	}
CALL PRINT_NUM	return 0;
JMP STRT	uva 10019:
	;UVA PRBLM 10071
	.MODEL SMALL
	.STACK 100H
EXIT:	.DATA
MOV AH,4CH	V DW 0
INT 21H	T DW 0
	.CODE
MAIN ENDP	MAIN PROC
DEFINE_SCAN_NUM	STRT:
DEFINE_PRINT_NUM	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
DEFINE_PRINT_NUM_UN\$	MOV DS,AX
END MAIN	
Uva 10071:	INCLUDE 'EMU8086.INC'
#include<stdio.h>	
int main()	;FOR USER INPUT
{	



DEFINE_SCAN_NUM	;COMPARE V AND T WITH 0;IF EQUAL THEN
DEFINE_PRINT_NUM	EXIT;OTHERWISE JUMP CALCULATION FOR
DEFINE_PRINT_NUM_UN	DISTANCE
	CMP AX,0
	JE EXIT
XOR CX,CX	JMP CALCULATION
XOR AX,AX	
XOR BX,BX ;O ASIGN ALL THE RESISTER	CMP T,0
XOR DX,DX	JE EXIT
MOV BX,2	JMP CALCULATION
; PRINTN "ENTER THE VALUE OF V"	CALCULATION:
	MUL T
CALL SCAN_NUM ; TAKE THE VALUE OF V	MUL BX
AND IT STORES THE VALUE IN CX	
printn	;PRINT " SHORTEST DISTANCE:"
MOV AX,CX	CALL PRINT_NUM
ADD V,AX ;V =V+AX	EXIT:
;PRINTN "ENTER THE VALUE OF T"	MOV AH,4CH
	INT 21H
CALL SCAN_NUM	JMP STRT
printN	RET
MOV T,CX	;
	END MAIN

```
//UVA PRBLM 10055
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    long long int c,a,b;
```

```
    while( scanf("%lld%lld",&a,&b)==2)
```

```
    {
```

```
        if(a>b)
```

```
        {
```

```
            c=a-b;
```

```
        }
```

```
    else
```

```
        c=b-a;
```

```
        printf("%lld\n",c);
```

```
    }
```

```
    return 0;
```

```
}
```

```
;
```

```
#UVA PRBLM 10055
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
A DW 0
```

```
N DW 0
```

```
B DW ?
```

```
R DW ?
```

```
K DW ?
```

```
.CODE
```

```
MAIN PROC
```

```
    STRT:
```

```
    MOV AX,@DATA
```

```
    MOV DS,AX    ;DATA INITIALIZATION
```

```
    INCLUDE 'EMU8086.INC'
```

```
    ;FOR USER INPUT
```

```
    XOR CX,CX
```

```
    XOR BX,BX    ;CLEAR REGISTER
```

```
    XOR AX,AX
```

```
    XOR DX,DX
```

```
    CALL SCAN_NUM ;DX = A
```

```
    MOV DX,CX
```

```
    PRINTN
```

```

CALL SCAN_NUM
MOV BX,CX    ;BX = B
PRINTN

CMP DX,BX    ;DX>BX
JG DIFFER    ;JUMP DIFFER
JMP DIFFER1

DIFFER:
    SUB DX,BX    ;DX=DX-BX
    MOV AX,DX    ;AX=DX
    JMP PRINT

DIFFER1:
    SUB BX,DX
    MOV AX,BX    ;BX=BX-DX
                ;AX=BX

PRINT:
    CALL PRINT_NUM ;PRINT AX

    PRINTN
JMP STRT

```

```

EXIT:
    MOV AH,4CH
    INT 21H

MAIN ENDP

DEFINE_SCAN_NUM
    DEFINE_PRINT_NUM
    DEFINE_PRINT_NUM_UN
END MAIN

Uva 10773:
    #include<stdio.h>
    #include<math.h>

int main()
{
    int n,i;
    double t1,t2,u,v,d;

    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        scanf("%lf%lf%lf",&d,&v,&u);
        if( v==0 || u==0 || v>=u)
            printf("Case %d: can't determine\n", i+1);
        else
        {
            t1 = d/u;

            t2 = (d/sqrt((u*u)-(v*v)));

```

printf("Case %d: %.3lf\n",i+1,t2-t1);	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
}	MOV DS,AX
return 0;}	
	INCLUDE 'EMU8086.INC'
;UVA PRBLM 10773	
.MODEL SMALL	XOR CX,CX
.STACK 100H	XOR AX,AX
.DATA	XOR BX,BX ;O ASIGN ALL THE RESISTER
N DW 0	XOR DX,DX
D DW 0	
u DW 0	call SCAN_NUM
v DW 0	mov n,cx
t1 DW 0	printn
t2 dw 0	
i dw 0	WHILE:
U1 DW 0	cmp n,1
V1 DW 0	je exit
DIFFER DW 0	
COUNTER DW 0	call scan_num
D3 DW 0	;MOV AX,CX ;D
B DW 0	MOV D,CX
A DW 0	printn
X DW 0	
ROOT DW 0	call scan_num
.CODE	mov DX,cx ;V
MAIN PROC	ADD V,DX
	printn
STRT:	

```
call scan_num
mov BX,cx
ADD U,BX    ;U
printn
```

```
MOV AX,U
MUL AX
MOV U1,AX
XOR AX,AX
```

```
L1:
    CMP V,0
    JE PRINT1
```

```
CALCULATION2:
    ;V*V
    MOV AX,V
    MUL AX
```

```
L2:
    CMP U,0
    JE PRINT1
```

```
;MOV V1,AX
; XOR AX,AX
```

```
L3:
    CMP V,BX
    JGE PRINT1
```

```
DIFFER1:
    SUB U1,AX
    XOR AX,AX
    MOV AX,U1
    MOV D3,AX
    XOR AX,AX
```

```
T1CAL:
    MOV AX,D
    DIV U
    MOV T1,AX
    XOR AX,AX
    XOR DX,DX
```

```
CALL SQRT
MOV BX,X
    MOV ROOT,BX
    MOV AX,D
    DIV ROOT
    MOV T2,AX
    XOR DX,DX
    XOR AX,AX
```

```
CALCULATION1:
    ;U*U
```

```
MOV CX,T2
```

SUB CX,T1	
MOV AX,CX	
CALL PRINT_NUM	SQRT PROC ;sqrt function strt
DEC N	MOV B,2
PRINTN	
JMP WHILE	;CALL SCAN_NUM ;take a number n
	;mov ah,1
	; int 21h
	MOV AX ,D3 ;ax==n
	ADD A,AX
	ADD X,AX
	mov N, CX ; x=n
	ADD BX,AX ;bx=n
PRINT1:	;DETERMINE SQRT ROOT OF GIVEN NUMBER
PRINTN "CANNOT DETERMINE"	DIV B ;n/2 porjonto loop continue hobe
DEC N	MOV COUNTER,AX ;from 1 to n/2
JMP WHILE	WHILE1:
	CMP COUNTER,1
	JE NEXT ; $(x+(n/x))/2$ formula n/2
EXIT:	porjonto continue korle sqrt pabo
	XOR AX,AX
MOV AH,4CH	ADD AX,A ;ax==n
INT 21H	xor dx,dx ;n/x
	DIV X
	;mov cx,ax ;x+ax
	ADD Ax,X
MAIN ENDP	xor dx,dx ;/2

DIV B	}
MOV X,ax	return 0;}
DEC COUNTER	;UVA 11172
JMP WHILE1	
NEXT:	.MODEL SMALL
RET	.STACK 100H
SQRT ENDP	.DATA
DEFINE_SCAN_NUM	A DW 0
DEFINE_PRINT_NUM	B DW 0
DEFINE_PRINT_NUM_UN	.CODE
END MAIN	MAIN PROC
#uva 11172	
#include<stdio.h>	
#include <math.h>	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
int main()	MOV DS,AX
{	
int t,a,b,c,i,d;	INCLUDE 'EMU8086.INC'
while(scanf("%d %d", &a, &b)==2)	
{	;FOR USER INPUT
if(a==0&&b==0)	
break;	DEFINE_SCAN_NUM
d=0;	DEFINE_PRINT_NUM
for(i=a;i<=b;i++)	DEFINE_PRINT_NUM_UN
{	
c=sqrt(i);	
if(c*c==i)	
d++;	XOR CX,CX
}	XOR AX,AX
printf("%dn", d);	XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX	JMP EXIT
PRINTN "ENTER THE VALUE OF A"	
CALL SCAN_NUM ; TAKE THE VALUE OF V AND IT STORES THE VALUE IN CX	PRINT3:
	PRINTN "="
	JMP EXIT
MOV AX,CX ;AX=CX=A	EXIT:
	MOV AH,4CH
	INT 21H
PRINTN "ENTER THE VALUE OF B"	
	END MAIN
CALL SCAN_NUM	#uva 10783
MOV BX,CX ;BX=CX=B	//UVA PRBLM 10783
	#include<stdio.h>
;COMPARE V AND T WITH 0;IF EQUAL THEN EXIT;OTHERWISE JUMP PRINT FOR PRINT	int main()
	{
CMP AX,BX	int sum,i,a,j,b,n;
JE PRINT3	scanf("%d",&n);
JG PRINT2	for(i=1;i<=n;i++){
JL PRINT1	scanf("%d%d",&a,&b);
	sum =0;
PRINT1:	for(j=a;j<=b;j++){
PRINTN "<"	{
JMP EXIT	
	if(j%2!=0)
PRINT2:	sum = sum+j;
PRINTN ">"	}
	printf("Case %d: %d\n",i,sum);



```

    }
    return 0;
}

;10783 - Odd Sum

.MODEL SMALL
.STACK 100H
.DATA
A DW 0
N DW 0
B DW 0
I DW 0
J DW 0
SUM1 DW 0
PUT DW 0
COUNT DW 0

.CODE

MAIN PROC
    MOV AX,@DATA ;DATA SEGMENT INITIALIZE
    MOV DS,AX

    INCLUDE 'EMU8086.INC'

;FOR USER INPUT

XOR AX,AX
XOR BX,BX

```

```

XOR CX,CX
XOR DX,DX

MOV B,2

CALL SCAN_NUM ;INPUT TESTCASE

MOV DX,CX
PRINTN

ADD N,DX
MOV I,1

FOR1:
    CMP N,1
    JE EXIT

CALL SCAN_NUM ; input a
MOV BX,CX
ADD PUT,BX

PRINTN

CALL SCAN_NUM ;input b
MOV CX,CX
printn

FOR:
    CMP PUT,CX
    JLE CAL

```

JMP PRINTF	
	MAIN ENDP
CAL:	
XOR DX,DX	DEFINE_SCAN_NUM
XOR AX,AX	DEFINE_PRINT_NUM
ADD AX,PUT	DEFINE_PRINT_NUM_UN
DIV B	END MAIN
CMP DX,0	#uva 10929
JNE SUM	#include<stdio.h>
INC PUT	int main()
XOR BX,BX	{
ADD BX,PUT	int a;
JMP FOR	int n;
	while(1)
SUM:	{
ADD COUNT,BX	scanf("%d",&a);
INC PUT	if(a==0)
XOR BX,BX	break;
ADD BX,PUT	else
JMP FOR	{
	if(a%11==0)
PRINTF:	printf("%d is a multiple of 11.\n",a);
XOR AX,AX	else
MOV AX,COUNT	printf("%d is not a multiple of 11.\n",a);
CALL PRINT_NUM	}
DEC N	
JMP FOR1	}
EXIT:	return 0;
	}

;UVA PRBLM 10929

.MODEL SMALL

.STACK 100H

.DATA

M DW 0

N DW 0

A DW 0

.CODE

MAIN PROC

STRT:

MOV AX,@DATA ;DATA SEGMENT INITIALIZE

MOV DS,AX

INCLUDE 'EMU8086.INC'

;FOR USER INPUT

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN

XOR CX,CX

XOR AX,AX

XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX

MOV N,11

CALL SCAN\_NUM

ADD A,CX

MOV M,CX

PRINTN

CMP M,0

JE EXIT

JMP CALCULATION

CALCULATION:

MOV AX,M

DIV N

CMP DX,0

JE PRINT1

JNE PRINT2

PRINT1:

XOR DX,DX

XOR AX,AX

MOV AX,A

CALL PRINT\_NUM

PRINTN "IS A MULTIPLE OF 11."

JMP STRT

PRINT2:

XOR DX,DX	if(c==sqrt((a*a)+(b*b)))
XOR AX,AX	printf("right\n");
MOV AX,A	else
CALL PRINT_NUM	printf("wrong\n");
PRINTN " :IS NOT A MULTIPLE OF 11."	
	}
	}
JMP STRT	return 0;
	}
EXIT:	;UVA PRBLM 11854
MOV AH,4CH	.MODEL SMALL
INT 21H	.STACK 100H
	.DATA
;	A DW 0
END MAIN	B DW 0
#uva 11854	C DW 0
//UVA PRBLM 11854	.CODE
#include<stdio.h>	MAIN PROC
#include<math.h>	
int main()	STRT:
{	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
int a,b,c,d,i,n;	MOV DS,AX
while(scanf("%d%d%d",&a,&b,&c)==2)	
{	INCLUDE 'EMU8086.INC'
if(a==0&&b==0&&c==0)	
break;	;FOR USER INPUT
else	
{	DEFINE_SCAN_NUM

DEFINE\_PRINT\_NUM  
DEFINE\_PRINT\_NUM\_UN

CMP BX,0  
JE EXIT  
XOR AX,AX

ADD BX,A

XOR CX,CX  
XOR AX,AX  
XOR BX,BX ;O ASIGN ALL THE RESISTER  
XOR DX,DX

;PRINTN "ENTER C"

CALL SCAN\_NUM

; PRINTN "ENTER A"

MOV AX,CX

ADD C,AX

CALL SCAN\_NUM

IMUL C

MOV DX,AX

MOV AX,CX

ADD A,AX

CMP DX,0

IMUL A

JE EXIT

MOV A,AX

XOR AX,AX

CMP A,0 ;A==0 THEN EXIT

JE EXIT

XOR AX,AX

CMP BX,DX ;C2==A2 \*B2

JE PRINTR

;PRINTN "ENTER B"

CALL SCAN\_NUM

PRINTN "WRONG"

MOV AX,CX

JMP EXIT

ADD B,AX

IMUL B

PRINTR:

MOV BX,AX

PRINTN "RIGHT"

EXIT:

```

MOV AH,4CH
    INT 21H

END MAIN
//UVA PRBLM 10300

```

```

#include<stdio.h>

int main()
{
    long long a,t,n,b,c,i,j,ans;
    while(scanf("%lld",&t)==1)
    {
        for(i=0;i<t;i++)
        {
            ans=0;
            scanf("%lld",&n);
            for(j=0;j<n;j++)
            {
                scanf("%lld%lld%lld",&a,&b,&c);
                ans+=a*c;
            }
            printf("%lld\n",ans);
        }
    }
    return 0;
}

;UVA PRBLM 10300

```

```

.MODEL SMALL

```

```

.STACK 100H

.DATA

V DW 0
T DW 0
ANS DW 0

.CODE

MAIN PROC

    STRT:

    MOV AX,@DATA ;DATA SEGMENT INITIALIZE
    MOV DS,AX

    INCLUDE 'EMU8086.INC'

    ;FOR USER INPUT

    DEFINE_SCAN_NUM
    DEFINE_PRINT_NUM
    DEFINE_PRINT_NUM_UN

    XOR CX,CX
    XOR AX,AX
    XOR BX,BX ;O ASIGN ALL THE RESISTER
    XOR DX,DX

    CALL SCAN_NUM

    MOV V,CX

```

```

PRINTN
COMP:
CMP V,0
JE PRINT
JMP CALCULATION
    CALCULATION:

;PRINTN "ENTER THE VALUE OF A"

CALL SCAN_NUM ; TAKE THE VALUE OF V
AND IT STORES THE VALUE IN CX

MOV AX,CX
PRINTN

; PRINTN "ENTER THE VALUE OF B"

CALL SCAN_NUM
MOV BX,CX
PRINTN

; PRINTN "ENTER THE VALUE OF C"

CALL SCAN_NUM
MOV DX,CX
PRINTN

IMUL DX ;AX=AX*DX=A*C

```

```

ADD ANS,AX
DEC V
JMP COMP

PRINT:

MOV AX,ANS
CALL PRINT_NUM
JMP EXIT

JMP STRT
EXIT:

MOV AH,4CH
INT 21H
JMP STRT
RET

;
END MAIN
//UVA PRBLM 12577

#include<stdio.h>
#include<string.h>
int main(){
char ary[6];
int counter=0;
while(1)
{
gets(ary);

```

if(ary[0]=='*')	MOV AH, 1 ;INPUT
{	INT 21H
break;	MOV BL, AL ;BL=AL
}	HUDDAI:
else if(ary[0]=='H')	INT 21H ;PRESS ENTER
{	CMP AL, 13
printf("Case %d: Hajj-e-Akbar\n",++counter);	JNE HUDDAI
}	CMP BL, '*'
if(ary[0]=='U')	JE ENDMAN
{	CMP BL, 'H'
printf("Case %d: Hajj-e-Asghar\n",++counter);	JE AKBAR
}	
}	LEA DX, UMRAH
return 0;	MOV AH, 9
}	INT 21H
;12577 - Hajj-e-Akbar	JMP START
ORG 100H	AKBAR:
	LEA DX, Hajj
.DATA	MOV AH, 9
Hajj DB 'Hajj-e-Akbar', 10, 13, '\$' ;10,13 NEW LINE	INT 21H
Umrah DB 'Hajj-e-Asghar', 10, 13, '\$'	JMP START
.CODE	ENDMAIN:
	RET
	#uva 12802
MAIN PROC	#include<stdio.h>
START:	long long palindrm(int n)



{	
int t,r;	MAIN PROC
r=0;	
	STRT:
t=n;	
while(t!=0)	XOR AX,AX
{	XOR BX,BX
r=r*10;	XOR CX,CX
r=r+t%10;	XOR DX,DX
t=t/10;	MOV D,2
}	
if(n==r)	CALL SCAN_NUM
return 1;	MOV AX,CX
return 0;	PRINTN
}	
;UVA 12802	ADD N,AX
INCLUDE 'EMU8086.INC'	MUL D
.MODEL SMALL	MOV AX,AX
.STACK 100H	CALL PRINT_NUM
.DATA	PRINTN
A DW 0	
N DW 0	
B DW 0	CALL PALLINDROM
D DW 2	MOV AX,A
VAR1 DW ?	
TEMP DW ?	CALL PRIME
.CODE	

MOV BX,B	AND CX,0     ;CX=0
	MOV CL,10D    ;CL=10
CHECK1:	
CMP AX,1	
JE CHECK2	NUMBER_SAVE:
JMP EXIT	MOV VAR1,BX
CHECK2:	
CMP BX,1	MOV AX,BX
JE STRT	XOR BX,BX
JMP EXIT	XOR DX,DX
	CHECK:
	DIV CL
	;MOV TEMP,AL
	MOV DL,AH
MAIN ENDP	XOR AH,AH
;FOR USER INPUT	MOV TEMP,AX
DEFINE_SCAN_NUM	MOV AX,BX
DEFINE_PRINT_NUM	MUL CL
DEFINE_PRINT_NUM_UN\$	ADD AX,DX
	MOV BX,AX
	MOV AX,TEMP
	CMP AX,0
	JE BREAK
PALLINDROM PROC	JMP CHECK
AND BX,0     ;BX=0	BREAK:

MOV DX,VAR1	
CMP BX,DX	
JE PRINT_SAME	call scan_num ;scan num from keyboard & it
JMP PRINT_DIFF	stores the value in cx
PRINT_SAME:	mov ax,cx
PRINTN	
XOR AX,AX	mov b,2
MOV A,1	div b
;MOV AX,A	
JMP EXIT	mov a,0
PRINT_DIFF:	cmp ax,a
PRINTN	je print1
XOR AX,AX	MOV B,0
MOV A,0	JMP EXIT1
;MOV AX,A	
EXIT:	
RET	
PALLINDROM ENDP	print1:
	MOV B,1
PRIME PROC	exit1:
mov ax,@data	RET
mov ds,ax ;data initialization	mov ah,4ch

int 21h	;uva 11364
end PRIME	.MODEL SMALL
	.STACK 100H
END MAIN	.DATA
;uva 11364	A DW 0
#include<stdio.h>	MX DW 0
#include<string.h>	MN DW 100
int main()	N DW 0
{	.CODE
int t,d,c,n;	MAIN PROC
scanf("%d",&t);	
while(t--)	
{	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
scanf("%d",&c);	MOV DS,AX
int mx=0,mn=100;	
while(c--)	INCLUDE 'EMU8086.INC'
{	
scanf("%d",&n);	;FOR USER INPUT
if(n>mx)	XOR CX,CX
mx=n;	XOR AX,AX
if(n<mn)	XOR BX,BX ;O ASIGN ALL THE RESISTER
mn=n;	XOR DX,DX
}	
d = 2*(mx-mn);	;PRINT "N:"
printf("%d\n",d);	
	CALL SCAN_NUM
}	MOV A,CX
return 0;	XOR CX,CX
}	PRINTN

```
    LOOP1:
    CMP A,0
    JE CALCULATION

    ;PRINT "NUM: "

    XOR AX,AX
    CALL SCAN_NUM
    PRINTN    ;;N==AX
    MOV AX,CX
    ADD N,AX
    XOR CX,CX
```

```
    CMP AX,MX
    JG L1
```

```
L1:
    MOV MX,AX
```

```
    XOR AX,AX
    MOV AX,N
    CMP AX,MN
    JL L2
```

```
L2:

    MOV MN,AX

    DEC A

    JMP LOOP1

CALCULATION:

    XOR CX,CX
    XOR AX,AX
    XOR BX,BX
    XOR DX,DX
```

```
    MOV CX,2
    ADD BX,MX
    ADD DX,MN
    SUB BX,DX
    MOV AX,BX
    IMUL CX
    MOV AX,CX
```

```
    CALL PRINT_NUM
    JMP EXIT
```

```
EXIT:
```

MOV AH,4CH	.MODEL SMALL
INT 21H	.STACK 100H
	.DATA
	V DW 0
MAIN ENDP	T DW 0
	.CODE
DEFINE_SCAN_NUM	MAIN PROC
DEFINE_PRINT_NUM	
DEFINE_PRINT_NUM_UN	
	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
END MAIN	MOV DS,AX
//UVA PRBLM 12372	
#include<stdio.h>	INCLUDE 'EMU8086.INC'
int main()	
{	;FOR USER INPUT
int t,l,w,h,count = 0;	
scanf("%d",&t);	DEFINE_SCAN_NUM
while(t--)	DEFINE_PRINT_NUM
{	DEFINE_PRINT_NUM_UN
scanf("%d%d%d",&l,&w,&h);	
if(l<=20 && w<=20 && h<=20)	
printf("Case %d: good\n",++count);	
else	XOR CX,CX
printf("Case %d: bad\n",++count);	XOR AX,AX
}	XOR BX,BX ;O ASIGN ALL THE RESISTER
return 0;	XOR DX,DX
}	
;UVA PRBLM 12372	CALL SCAN_NUM

```

MOV AX,CX

PRINTN    ;SCAN L

; XOR CX,CX

CALL SCAN_NUM

MOV BX,CX    ;SCAN W

PRINTN

;XOR CX,CX

CALL SCAN_NUM    ;SCAN H

PRINTN

MOV DX,CX

CMP AX,20    ;IF CONDITION

JLE LEVEL1

JMP PRINT2

LEVEL1:

    CMP BX,20

    JLE LEVEL2

    JMP PRINT2

LEVEL2:

    CMP DX,20

    JLE PRINT1

    JMP PRINT2

```

```

PRINT1:

    PRINTN " CASE 1: GOOD "

    JMP EXIT

PRINT2:

    PRINTN " CASE 1: BAD "

EXIT:

    MOV AH,4CH

    INT 21H

    RET

;

END MAIN

#uva 11479

#include<stdio.h>

int main()

{

    long int t,a,b,c,i;

    while(scanf("%ld",&t)==1)

    {

        i=1;

        while(i<=t)

        {

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

            if((a+b)<=c || (b+c)<=a || (c+a)<=b)

```

```

printf("Case %ld: Invalid\n",i);
else if(a<=0 || b<=0 || c<=0)
printf("Case %ld: Invalid\n",i);
else if(a==b && b==c)
printf("Case %ld: Equilateral\n",i);
else if(a==b || b==c || c==a)
printf("Case %ld: Isosceles\n",i);
else
printf("Case %ld: Scalene\n",i);
i++;
}

```

```

}

```

```

return 0;

```

```

}

```

```

;#UVA PRBLM 11479

```

```

.MODEL SMALL

```

```

.STACK 100H

```

```

.DATA

```

```

A DW 0

```

```

B DW 0

```

```

C DW 0

```

```

SUM1 DW 0

```

```

SUM2 DW 0

```

```

SUM3 DW 0

```

```

N DW 0

```

```

M DW 0

```

```

P DW 0

```

```

.CODE

```

```

MAIN PROC

```

```

    STRT:

```

```

    MOV AX,@DATA

```

```

    MOV DS,AX    ;DATA INITIALIZATION

```

```

    INCLUDE 'EMU8086.INC'

```

```

    ;FOR USER INPUT

```

```

    XOR CX,CX

```

```

    XOR BX,BX    ;CLEAR REGISTER

```

```

    XOR AX,AX

```

```

    xor dx,dx

```

```

    CALL SCAN_NUM

```

```

    MOV AX,CX    ;INPUT A

```

```

    ADD A,AX

```

```

    ADD N,AX

```

```

    PRINTN

```

```

    CALL SCAN_NUM

```

```

    MOV BX,CX

```

```

    ADD B,BX

```

```

    ADD M,BX    ;INPUT B

```



PRINTN

CALL SCAN\_NUM

MOV CX,CX

ADD C,CX

ADD P,CX ;INPUT C

PRINTN

S:

XOR AX,AX

MOV AX,A

ADD AX,B ;A+B

MOV SUM1,AX

S1:

XOR AX,AX

MOV AX,B

ADD AX,C ;B+C

MOV SUM2,AX

S3:

XOR AX,AX

MOV AX,C

ADD AX,N ;C+A

MOV SUM3,AX

L1:

XOR AX,AX

MOV AX,SUM1

CMP AX,P

JLE PRINT1

L2:

XOR AX,AX

MOV AX,SUM2

CMP AX,M

JLE PRINT1

L3:

XOR AX,AX

MOV AX,SUM3

CMP AX,N

JLE PRINT1

L4:

CMP M,0

JLE PRINT1

L5:

CMP N,0

JLE PRINT1

L6:

CMP P,0

JLE PRINT1

CHECK:

XOR AX,AX

ADD AX,M ;A==B

CMP AX,N	
JE CHECK1	PRINT1:
	PRINTN " INVALID"
CHECK1:	JMP EXIT
XOR AX,AX	
ADD AX,N ;B==C	PRINT2:
CMP AX,P	PRINTN "EQUILATERAL"
JE PRINT2	JMP EXIT
CHECK12:	PRINT3:
XOR AX,AX	PRINTN "ISOSCELES"
ADD AX,M ;A==B	JMP EXIT
CMP AX,N	
JE PRINT3	PRINTN "SCALENE"
CHECK13:	
XOR AX,AX	
ADD AX,N ;B==C	
CMP AX,P	EXIT:
JE PRINT3	MOV AH,4CH
	INT 21H
	MAIN ENDP
CHECK2:	
XOR AX,AX	DEFINE_SCAN_NUM
ADD AX,P	DEFINE_PRINT_NUM
CMP AX,M	DEFINE_PRINT_NUM_UN\$
JE PRINT3	END MAIN
	//UVA PRBLM 10079

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int p;
```

```
    long long int n;
```

```
    while(scanf("%lld",&n))
```

```
    {
```

```
        if(n<0)
```

```
            break;
```

```
        printf("%lld\n",1+ (n*(n+1)/2));
```

```
    }
```

```
    return 0;
```

```
}
```

```
;UVA PRBLM 10079
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
A DW 0
```

```
N DW 0
```

```
B DW ?
```

```
R DW ?
```

```
K DW ?
```

```
.CODE
```

```
MAIN PROC
```

```
    STRT:
```

```
    MOV AX,@DATA
```

```
    MOV DS,AX ;DATA INITIALIZATION
```

```
INCLUDE 'EMU8086.INC'
```

```
;FOR USER INPUT
```

```
XOR CX,CX
```

```
XOR BX,BX ;CLEAR REGISTER
```

```
XOR AX,AX
```

```
XOR DX,DX
```

```
MOV BX,2
```

```
CALL SCAN_NUM ;SCAN N
```

```
PRINT
```

```
MOV AX,CX
```

```
ADD A,AX
```

```
CMP AX,0
```

```
JE EXIT ;N<0 BREAK
```

```
ADD AX,1 ;ELSE N+1
```

```
MUL A ;N*N+1
```

```
DIV BX ;/2
```

ADD AX,1 ;+1	break;
PRINTN " "	scanf("%d%d",&n,&m);
	for(i=0; i<k; i++)
	{
CALL PRINT_NUM	scanf("%d%d",&x,&y);
	if(n==x    m==y)
	printf("divisa\n");
JMP STRT	else if(x>n && y>m)
	printf("NE\n");
	else if(x<n&&y>m)
	{
EXIT:	printf("NO\n");
MOV AH,4CH	}
INT 21H	else if(x<n&&y<m)
	{
MAIN ENDP	printf("SO\n");
	}
DEFINE_SCAN_NUM	else if(x>n&&y<m)
DEFINE_PRINT_NUM	{
DEFINE_PRINT_NUM_UN\$	printf("SE\n");
END MAIN	}
//UVA PRBLM 11498	
#include<stdio.h>	}
int main()	}
{	return 0;
int n,m,i,x,y,k;	}
while(scanf("%d",&k)==1)	;
{	
if(k==0)	

.MODEL SMALL

.STACK 100H	MOV K,CX
.DATA	PRINTN ;CX = K
A DW 0	
N DW 0	CMP K,0
M DW 0	JE EXIT
I DW 0	
K DW 0	CALL SCAN_NUM
X DW 0	MOV N,CX
Y DW 0	PRINTN
.CODE	
MAIN PROC	CALL SCAN_NUM
STRT:	MOV M,CX
MOV AX,@DATA	PRINTN
MOV DS,AX ;DATA INITIALIZATION	
	MOV DX,0 ;I=0=DX
INCLUDE 'EMU8086.INC'	LOOP1:
;FOR USER INPUT	CMP DX,K
	JG EXIT
	CALL SCAN_NUM
	MOV AX,CX ;X=AX
XOR CX,CX	PRINTN
XOR BX,BX ;CLEAR REGISTER	
XOR AX,AX	CALL SCAN_NUM
XOR DX,DX	MOV BX,CX ;Y=BX
CALL SCAN_NUM	CMP AX,N
	JE CHECK1 ;X== N

```
CMP AX,N
JG CHECK2 ;X>N

CMP AX,N ;X<N
JL CHECK3
```

```
CMP AX,N
JL CHECK4 ;X<N
```

```
CHECK1:
    CMP BX,M ;Y==M
    JE PRINT1
```

```
CHECK2:
    CMP BX,M
    JG PRINT2 ;Y>M
```

```
CHECK3:
    CMP BX,M
    JG PRINT3
```

```
CHECK4:
    CMP BX,M
    JL PRINT4
```

```
PRINT1:
    PRINTN "DIVISA"
    INC DX
    JMP LOOP1
```

```
PRINT2:
    PRINTN "NE"
    INC DX
    JMP LOOP1
```

```
PRINT3:
    PRINTN "NO"
    INC DX
    JMP LOOP1
```

```
PRINT4:
    PRINTN "SO"
    INC DX
```

```
JMP LOOP1
```

```
EXIT:
    MOV AH,4CH
    INT 21H
```

```
MAIN ENDP
```

DEFINE_SCAN_NUM	.CODE
DEFINE_PRINT_NUM	MAIN PROC
DEFINE_PRINT_NUM_UN\$	
END MAIN	
//UVA PRBLM 12646	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
#include <stdio.h>	MOV DS,AX
int main()	
{	INCLUDE 'EMU8086.INC'
int a,b,c;	
while(scanf("%d %d %d", &a, &b, &c) == 3)	;FOR USER INPUT
{	
if(a == b && b == c)	XOR CX,CX
printf("*\n");	XOR AX,AX
else if(a != b && b == c)	XOR BX,BX ;O ASIGN ALL THE RESISTER
printf("A\n");	XOR DX,DX
else if(a != b && a == c)	
printf("B\n");	CALL SCAN_NUM
else if(a == b && a != c)	;SCAN A
printf("C\n");	MOV AX,CX
}	PRINTN
return 0;	; XOR CX,CX
}	
;UVA PRBLM 12646	
	CALL SCAN_NUM
	;SCAN B
.MODEL SMALL	MOV BX,CX
.STACK 100H	PRINTN
.DATA	;XOR CX,CX
V DW 0	
T DW 0	

CALL SCAN\_NUM ;SCAN C

PRINTN

MOV DX,CX

LEVEL1:

CMP AX,BX

JE LEVEL3 ;AX==BX

JNE LEVEL8

LEVEL2:

CMP AX,DX ;AX==DX

JE LEVEL4

LEVEL3:

CMP BX,DX ;BX==DX

JE PRINT1

LEVEL4:

CMP AX,BX

JNE PRINT3 ;AX!=BX

LEVEL5:

CMP AX,DX ;AX!=DX

JNE LEVEL7

LEVEL6:

CMP BX,DX

JE PRINT2 ;BX==DX

LEVEL7:

CMP AX,BX

JE PRINT4 ;AX==BX

LEVEL8:

CMP AX,DX ;AX==DX

JE PRINT3

PRINT1:

PRINTN "\*"

JMP EXIT

PRINT2:

PRINTN "A"

JMP EXIT

PRINT3:

PRINTN "B"

JMP EXIT

PRINT4:

PRINTN "C"

JMP EXIT

EXIT:

MOV AH,4CH



INT 21H	}
	;uva 12952
	;#UVA PRBLM 12952
MAIN ENDP	
	.MODEL SMALL
DEFINE_SCAN_NUM	.STACK 100H
DEFINE_PRINT_NUM	.DATA
DEFINE_PRINT_NUM_UN\$	A DW 0
	N DW 0
	B DW ?
;	R DW ?
END MAIN	K DW ?
	.CODE
	MAIN PROC
	STRT:
//uva 12952	MOV AX,@DATA
#include<stdio.h>	MOV DS,AX ;DATA INITIALIZATION
int main()	
{	
int a,b;	INCLUDE 'EMU8086.INC'
while( scanf("%d%d",&a,&b)!=EOF)	;FOR USER INPUT
{	
if(a>=b)	
printf("%d\n",a);	
else	
printf("%d\n",b);	XOR CX,CX
}	XOR BX,BX ;CLEAR REGISTER
return 0;	XOR AX,AX
	XOR DX,DX

```

CALL SCAN_NUM ;DX = A
MOV DX,CX
PRINTN

CALL SCAN_NUM
MOV BX,CX ;BX = B
PRINTN

CMP DX,BX
JGE L1

L2:
MOV AX,BX
CALL PRINT_NUM
JMP EXIT

L1:
MOV AX,DX
CALL PRINT_NUM

EXIT:
MOV AH,4CH
INT 21H

MAIN ENDP

```

```

DEFINE_SCAN_NUM
DEFINE_PRINT_NUM
DEFINE_PRINT_NUM_UN
END MAIN

//UVA PRBLM 10696
#include<stdio.h>

int main()
{
    int n,N=0;
    while(scanf("%d",&n)==1&&n!=0)
    {
        if(n <= 101)
            printf("f91(%d) = 91\n",n);
        else
            printf("f91(%d) = %d\n",n,n-10);
    }
    return 0;
}

//;UVA PRBLM 10696

.MODEL SMALL
.STACK 100H
.DATA
A DW 0
B DW 0
C DW 0
.CODE
MAIN PROC

```

	PRINT "("
MOV AX,@DATA ;DATA SEGMENT INITIALIZE	
MOV DS,AX	CALL PRINT_NUM
	PRINT ")"
INCLUDE 'EMU8086.INC'	PRINT "= 91"
	PRINTN " "
;FOR USER INPUT	JMP EXIT
DEFINE_SCAN_NUM	CALCULATION:
DEFINE_PRINT_NUM	;ADD A,AX
DEFINE_PRINT_NUM_UN\$	; PUSH AX
	MOV BX,AX
	SUB BX,10
	MOV AX,BX
XOR CX,CX	PRINT "F91 ="
XOR AX,AX	PRINT "("
XOR BX,BX ;O ASIGN ALL THE RESISTER	CALL PRINT_NUM
XOR DX,DX	PRINT ")"
	;POP CX
call SCAN_NUM ;SCAN N	PRINTN " "
MOV AX,CX	;POP BX
PRINTN	
	EXIT:
CMP AX,101 ;CHECK IF CONDITION	
JLE PRINT	MOV AH,4CH
JMP CALCULATION	INT 21H
PRINT:	;
PRINT "F91"	END MAIN

```
//UVA PRBLM 11150
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
int a,n;
```

```
while(scanf("%d",&n)==1)
```

```
{
```

```
a=n;
```

```
while(n>=3)
```

```
{
```

```
a=a+(n/3);
```

```
n=(n/3)+(n%3);
```

```
}
```

```
if(n==2)
```

```
{
```

```
a++;
```

```
}
```

```
printf("%d\n",a);
```

```
}
```

```
return 0;
```

```
}
```

```
;UVA PRBLM 11150
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
A DW 0
```

```
T DW 0
```

```
.CODE
```

```
MAIN PROC
```

```
MOV AX,@DATA ;DATA SEGMENT INITIALIZE
```

```
MOV DS,AX
```

```
INCLUDE 'EMU8086.INC'
```

```
;FOR USER INPUT
```

```
XOR CX,CX
```

```
XOR AX,AX
```

```
XOR BX,BX ;O ASIGN ALL THE RESISTER
```

```
CALL SCAN_NUM ;N==AX
```

```
MOV AX,CX
```

```
XOR CX,CX
```

```
PRINTN
```

```
MOV A,AX
```

```
MOV BX,3 ;BX ==3
```

```
WHILE:
```

```
CMP AX,BX
```

```
JGE CALCULATION
```

```
JMP IF
```

CALCULATION:

XOR DX, DX

DIV BX

ADD A,AX

ADD AX,DX

JMP WHILE

IF:

CMP AX,2

JE LOOP1

JMP PRINT

LOOP1:

INC A

JMP PRINT

PRINT:

MOV AX,A

CALL PRINT\_NUM

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN\$

END MAIN

//uva 12917

#include<stdio.h>

int main()

{

int x,y,z;

while(scanf("%d%d%d",&x,&y,&z)==3)

{

if(x<=(z-y))

printf("Props win!\n");

else

printf("Hunters win!\n");

}

return 0;

}

;\$UVA PRBLM 12917

.MODEL SMALL

.STACK 100H

.DATA

A DW 0

N DW 0

B DW ?	CALL SCAN_NUM
R DW ?	MOV AX,CX ;Z INPUT
K DW ?	PRINTN
.CODE	
MAIN PROC	SUB AX,BX
STRT:	MOV A,AX ;Z-Y
MOV AX,@DATA	
MOV DS,AX ;DATA INITIALIZATION	CMP DX,A
	JLE PRINT12
INCLUDE 'EMU8086.INC'	PRINTN "Hunters win!"
;FOR USER INPUT	jmp strt
	PRINT12:
	printn "Props win!"
XOR CX,CX	jmp strt
XOR BX,BX ;CLEAR REGISTER	
XOR AX,AX	
XOR DX,DX	EXIT:
	MOV AH,4CH
CALL SCAN_NUM	INT 21H
MOV DX,CX ;X INPUT	
PRINTN	MAIN ENDP
CALL SCAN_NUM	DEFINE_SCAN_NUM
MOV BX,CX ;Y INPUT	DEFINE_PRINT_NUM
PRINTN	DEFINE_PRINT_NUM_UN\$
	END MAIN

```
//UVA PRBLM 12992
```

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
    int n,m,count = 1;
```

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

```
    while(n--)
```

```
    {
```

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

```
        printf("Case #%%d: %%d\\n",count++,2*m-1);
```

```
    }
```

```
    return 0;
```

```
}
```

```
;
```

```
#UVA PRBLM 12992
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
A DW 0
```

```
N DW 0
```

```
B DW 0
```

```
I DW 0
```

```
J DW 0
```

```
.CODE
```

```
MAIN PROC
```

```
    STRT:
```

```
    MOV AX,@DATA
```

```
    MOV DS,AX ;DATA INITIALIZATION
```

```
INCLUDE 'EMU8086.INC'
```

```
;FOR USER INPUT
```

```
XOR CX,CX
```

```
XOR BX,BX ;CLEAR REGISTER
```

```
XOR AX,AX
```

```
XOR DX,DX
```

```
MOV B,2
```

```
CALL SCAN_NUM
```

```
MOV BX,CX
```

```
mov a,0
```

```
PRINTN
```

```
WHILE:
```

```
    CMP BX,A
```

```
    JE EXIT
```

```
    CALL SCAN_NUM
```

```
    MOV AX,CX
```

```
    PRINTN
```

```
    MUL B
```

```
    SUB AX,1
```

```
    MOV DX,AX
```

INC A	while(scanf("%d",&n)!=EOF)
	{
PRINT "Case #"	count =0;
MOV AX,A	
CALL PRINT_NUM	
PRINT ":"	for(i=0;i<5;i++){
MOV AX,DX	scanf("%d",&m);
CALL PRINT_NUM	if(n==m)
PRINTN	count++;
JMP WHILE	}
EXIT:	printf("%d\n",count);
MOV AH,4CH	}
INT 21H	return 0;
	}
	;#UVA PRBLM 13012
MAIN ENDP	
	.MODEL SMALL
DEFINE_SCAN_NUM	.STACK 100H
DEFINE_PRINT_NUM	.DATA
DEFINE_PRINT_NUM_UN\$	A DW 0
END MAIN	N DW 0
//UVA PRBLM 13012	B DW 0
#include<stdio.h>	I DW 0
int main()	J DW 0
{	.CODE
int n,m,i,count;	MAIN PROC



STRT:

MOV AX,@DATA

MOV DS,AX ;DATA INITIALIZATION

CMP DX,BX

JE LOOP1

JMP CAL

INCLUDE 'EMU8086.INC'

;FOR USER INPUT

LOOP1:

INC J

JMP CAL

XOR CX,CX

XOR BX,BX ;CLEAR REGISTER

XOR AX,AX

XOR DX,DX

PRINT1:

MOV AX,J

;PRINTN

CALL PRINT\_NUM

PRINTN

call scan\_num

MOV BX,CX

PRINTN

MOV A,0

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

CAL:

CMP BX,A

JE PRINT1

INC A

CALL SCAN\_NUM

MOV DX,CX

PRINTN

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN\$

END MAIN

//uva 13025

#include<stdio.h>

```

int main()
{
    printf("May 29, 2013 Wednesday\n");
    return 0;}
;#UVA PRBLM 13025

```

```

.MODEL SMALL

```

```

.STACK 100H

```

```

.DATA

```

```

A DW 0

```

```

N DW 0

```

```

B DW ?

```

```

R DW ?

```

```

K DW ?

```

```

.CODE

```

```

MAIN PROC

```

```

    STRT:

```

```

    MOV AX,@DATA

```

```

    MOV DS,AX    ;DATA INITIALIZATION

```

```

    INCLUDE 'EMU8086.INC'

```

```

;FOR USER INPUT

```

```

    XOR CX,CX

```

```

    XOR BX,BX    ;CLEAR REGISTER

```

```

XOR AX,AX

```

```

XOR DX,DX

```

```

PRINTN " May 29,2013 Wednesday"

```

```

EXIT:

```

```

MOV AH,4CH

```

```

INT 21H

```

```

MAIN ENDP

```

```

DEFINE_SCAN_NUM

```

```

DEFINE_PRINT_NUM

```

```

DEFINE_PRINT_NUM_UN

```

```

END MAIN

```

```

//uva 12149

```

```

#include<stdio.h>

```

```

int main()

```

```

{

```

```

    int n,s;

```

```

    while(scanf("%d",&n)==1)

```

```

    {

```

```

        if(n==0)

```

```

            break;

```

```

            s=(n*(n+1)*(2*n+1))/6;

```

```

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

```

```

    }

```

```
    return 0;
}
;UVA PRBLM 12149
```

```
.MODEL SMALL
```

```
.STACK 100H
```

```
.DATA
```

```
N DW 0
```

```
N1 DW 0
```

```
TEMP DW 0
```

```
N2 DW 0
```

```
A DW 0
```

```
B DW 0
```

```
I DW 0
```

```
.CODE
```

```
MAIN PROC
```

```
    STRT:
```

```
    MOV AX,@DATA ;DATA SEGMENT INITIALIZE
```

```
    MOV DS,AX
```

```
    INCLUDE 'EMU8086.INC'
```

```
    ;FOR USER INPUT
```

```
    DEFINE_SCAN_NUM
```

```
    DEFINE_PRINT_NUM
```

```
    DEFINE_PRINT_NUM_UNS
```

```
    XOR CX,CX
```

```
    XOR AX,AX
```

```
    XOR BX,BX ;O ASIGN ALL THE RESISTER
```

```
    XOR DX,DX
```

```
    MOV A,2
```

```
    MOV B,6
```

```
    CALL SCAN_NUM
```

```
    MOV AX,CX
```

```
    ADD N,AX
```

```
    ADD N1,AX
```

```
    PRINTN
```

```
    CMP AX,0
```

```
    JE EXIT
```

```
    FIRSTCAL: ;N*(N+1)
```

```
    ADD N,1
```

```
    MUL N
```

```
    mov bx,AX
```

```
    SECONDCAL:
```

```
        MOV AX,N1
```

```
        MUL A ;N*2+1
```

ADD AX,1	int m,n,c;
	while(scanf("%d%d",&m,&n)==2)
	printf("%d\n",m*n-1);
THIRDCAL:	return 0;}
MUL BX	;UVA PRBLM 10970
FINALCAL:	.MODEL SMALL
	.STACK 100H
DIV B	.DATA
CALL PRINT_NUM	M DW 0
PRINTN	N DW 0
XOR AX,AX	.CODE
	MAIN PROC
JMP STRT	
	STRT:
	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
	MOV DS,AX
EXIT:	
	INCLUDE 'EMU8086.INC'
MOV AH,4CH	
INT 21H	;FOR USER INPUT
END MAIN	DEFINE_SCAN_NUM
	DEFINE_PRINT_NUM
	DEFINE_PRINT_NUM_UN\$
//UVA PRBLM 10970	
#include<stdio.h>	
int main()	
{	XOR CX,CX

XOR AX,AX

XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX

PRINTN "ENTER THE VALUE OF M"

CALL SCAN\_NUM ; TAKE THE VALUE OF V  
AND IT STORES THE VALUE IN CX

MOV AX,CX ;AX=CX=M

PRINTN "ENTER THE VALUE OF N "

CALL SCAN\_NUM

MOV BX,CX ;BX=CX=N

SUB BX,1 ;CX= N-1=BX-1

IMUL BX

PRINTN " BIG CHOLCATE PICES:"

CALL PRINT\_NUM

JMP STRT

EXIT:

MOV AH,4CH

INT 21H

;

END MAIN

//uva 10110

#include<stdio.h>

#include<math.h>

int main()

{

long long int a,b,c,d;

while(scanf("%lld",&a)==1)

{

if(a==0)

break;

b = sqrt(a);

c = b\*b;

if(c==a)

printf("yes\n");

else

```

        printf("no\n");
    }
    return 0;
}

```

;UVA PRBLM 10110

.MODEL SMALL

.STACK 100H

.DATA

N DW 0

X DW 0

A DW 0

B DW 0

C DW 0

COUNTER DW 0

.CODE

MAIN PROC

STRT:

MOV AX,@DATA ;DATA SEGMENT INITIALIZE

MOV DS,AX

INCLUDE 'EMU8086.INC'

XOR CX,CX

XOR AX,AX

XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX

CALL SQRT ;call sqrt function

MOV AX,BX

MUL AX

; MOV C,AX

CMP N,AX

JE PRINT1

PRINTN "NO"

JMP EXIT

PRINT1:

PRINTN "YES"

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

SQRT PROC ;sqrt function strt

MOV B,2d

CALL SCAN_NUM ;take a number n	RET
;mov ah,1	SQRT ENDP
; int 21h	;FOR USER INPUT
MOV AX ,CX ;ax==n	DEFINE_SCAN_NUM
ADD A,AX	DEFINE_PRINT_NUM
ADD X,AX	DEFINE_PRINT_NUM_UN\$
mov N, CX ; x=n	
ADD BX,AX ;bx=n	
;DETERMINE SQRT ROOT OF GIVEN NUMBER	;
	END MAIN
DIV B ;n/2 porjonto loop continue hobe	
MOV COUNTER,AX ;from 1 to n/2	//UVA PRBLM 10346
WHILE:	#include<stdio.h>
CMP COUNTER,1	int main()
JE NEXT ;(x+(n/x))/2 formula n/2	{
porjonto continue korle sqrt pabo	int a,n,k;
XOR AX,AX	while(scanf("%d %d",&n,&k)==2 && k>1)
ADD AX,A ;ax==n	{
xor dx,dx ;n/x	a=n;
DIV BX	while(n>=k)
mov cx,ax ;x+ax	{
ADD ax,BX	a=a+(n/k);
xor dx,dx ;/2	n=(n/k)+(n%k);
DIV B	}
MOV BX,ax	printf("%d\n",a);
DEC COUNTER	}
JMP WHILE	return 0;
NEXT:	}

;UVA PRBLM 10346

XOR DX,DX

.MODEL SMALL

PRINTN "ENTER VALUE N:"

.STACK 100H

.DATA

CALL SCAN\_NUM ;IT STORES THE VALUE IN  
CX

A DW 0

N DW 0

MOV ax,CX ;SCANF THE VALUE OF N

B DW ?

ADD N,AX

R DW ?

ADD A,AX

K DW ?

PRINTN "ENTER VALUE K:"

.CODE

MAIN PROC

MOV AX,@DATA

CALL SCAN\_NUM

MOV DS,AX ;DATA INITIALIZATION

MOV CX,CX ;SCANF THE VALUE OF K

INCLUDE 'EMU8086.INC'

WHILE:

;FOR USER INPUT

CMP N,CX

JGE LOOP1

JMP POPOUT

DEFINE\_SCAN\_NUM

LOOP1:

DEFINE\_PRINT\_NUM

DIV CX

DEFINE\_PRINT\_NUM\_UN\$

MOV B,AX

;XOR B,B

MOV R,DX

;XOR R,R

XOR CX,CX

ADD AX,B

XOR BX,BX

XOR AX,AX



	}
ADD DX,B	else if(a<b)
MOV N,DX	printf("<\n");
	else
JMP WHILE	printf("=\n");}
	return 0;
POPOUT:	}
;	;UVA 11172
;MOV AH,2	
;ADD AX,48	.MODEL SMALL
;MOV DX,AX	.STACK 100H
call print_num	.DATA
;INT 21H	A DW 0
	B DW 0
; MAIN ENDP	.CODE
END MAIN	MAIN PROC
//UVA PRVLM 11172	
#include<stdio.h>	MOV AX,@DATA ;DATA SEGMENT INITIALIZE
int main()	MOV DS,AX
{	
int n,a,b;	INCLUDE 'EMU8086.INC'
scanf("%d",&n);	
	;FOR USER INPUT
while(scanf("%d%d",&a,&b)!=n){	
if(a>b)	DEFINE_SCAN_NUM
{	DEFINE_PRINT_NUM
printf(">\n");	DEFINE_PRINT_NUM_UN\$

```

XOR CX,CX

XOR AX,AX

XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX

PRINTN "ENTER THE VALUE OF A"

CALL SCAN_NUM ; TAKE THE VALUE OF V
AND IT STORES THE VALUE IN CX

MOV AX,CX ;AX=CX=A

PRINTN "ENTER THE VALUE OF B"

CALL SCAN_NUM

MOV BX,CX ;BX=CX=B

;COMPARE V AND T WITH 0;IF EQUAL THEN
EXIT;OTHERWISE JUMP PRINT FOR PRINT

CMP AX,BX

JE PRINT3

JG PRINT2

JL PRINT1

```

```

PRINT1:

PRINTN "<"

JMP EXIT

PRINT2:

PRINTN ">"

JMP EXIT

PRINT3:

PRINTN "="

JMP EXIT

EXIT:

MOV AH,4CH

INT 21H

END MAIN

//uva 11461

#include<stdio.h>

#include<math.h>

int main()

{

int a,b,n,c,i;

while(scanf("%d%d",&a,&b)==2){

int d=0;

if(a==0&&b==0)

break;

```

else	I DW 0
{	D DW 0
	COUNTER DW 0
for(i=a;i<=b;i++)	.CODE
{	MAIN PROC
c=sqrt(i);	STRT:
if(c*c==i)	MOV AX,@DATA
d++;	MOV DS,AX ;DATA INITIALIZATION
}}	
printf("%d\n",d);}	INCLUDE 'EMU8086.INC'
	;FOR USER INPUT
return 0;	
}	
;/#UVA PRBLM 11461	XOR CX,CX
	XOR BX,BX ;CLEAR REGISTER
.MODEL SMALL	XOR AX,AX
.STACK 100H	XOR DX,DX
.DATA	
A DW 0	CALL SCAN_NUM
N DW 0	MOV BX,CX
B DW 0	ADD COUNT1,BX ;A
X DW 0	PRINTN
C DW 0	
COUNT1 DW 0	CALL SCAN_NUM
COUNT2 DW 0	MOV DX,CX

ADD COUNT2,DX ;B

PRINTN

CMP BX,0

JE EXIT

CMP DX,0

JE EXIT

MOV I,BX ;I==A

FORLOOP:

CMP I,DX

JGE PRINT1

CALL SQRT

MOV Cx,bX

; INC I

MOV AX,Cx

MUL AX

CMP AX,I

JE L1

INC I

JMP FORLOOP

L1:

INC D

INC I

JMP FORLOOP

PRINT1:

MOV AX,D

CALL PRINT\_NUM

PRINTN

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

SQRT PROC ;sqrt function strt

MOV B,2

; CALL SCAN\_NUM ;take a number n

;mov ah,1

; int 21h

XOR AX,AX

MOV AX ,I ;ax==n

ADD A,AX

ADD X,AX

mov N, CX ; x=n

```

ADD BX,AX      ;bx=n
;DETERMINE SQRT ROOT OF GIVEN NUMBER

DIV B          ;n/2 porjonto loop continue hobe
MOV COUNTER,AX ;from 1 to n/2
XOR DX,DX
WHILE:
    CMP COUNTER,1
    JE NEXT     ;(x+(n/x))/2 formula n/2
                ;porjonto continue korle sqrt pabo
    XOR AX,AX
    ADD AX,A     ;ax==n
    xor dx,dx    ;n/x
    DIV bX
    mov cx,ax    ;x+ax
    ADD ax,bX
    xor dx,dx    ;/2
    DIV B
    MOV bX,ax
    DEC COUNTER
    JMP WHILE
NEXT:
RET
SQRT ENDP

```

```

DEFINE_SCAN_NUM
DEFINE_PRINT_NUM

```

```

DEFINE_PRINT_NUM_UNS
END MAIN

//UVA PRBLM 10783
#include<stdio.h>
int main()
{
    int sum,i,a,j,b,n;
    scanf("%d",&n);
    for(i=1;i<=n;i++){
        scanf("%d%d",&a,&b);
        sum =0;
        for(j=a;j<=b;j++)
        {
            if(j%2!=0)
                sum = sum+j;
        }
        printf("Case %d: %d\n",i,sum);
    }
    return 0;
}
;10783 - Odd Sum

.MODEL SMALL
.STACK 100H
.DATA
A DW 0
N DW 0
B DW 0

```

I DW 0	CMP N,1
J DW 0	JE EXIT
SUM1 DW 0	
PUT DW 0	CALL SCAN_NUM ; input a
COUNT DW 0	MOV BX,CX
	ADD PUT,BX
.CODE	
	PRINTN
MAIN PROC	
MOV AX,@DATA ;DATA SEGMENT INITIALIZE	CALL SCAN_NUM ;input b
MOV DS,AX	MOV CX,CX
	printn
INCLUDE 'EMU8086.INC'	
	FOR:
;FOR USER INPUT	
XOR AX,AX	CMP PUT,CX
XOR BX,BX	JLE CAL
XOR CX ,CX	JMP PRINTF
XOR DX,DX	
	CAL:
MOV B,2	XOR DX,DX
	XOR AX,AX
CALL SCAN_NUM ;INPUT TESTCASE	ADD AX,PUT
MOV DX,CX	DIV B
PRINTN	CMP DX,0
ADD N,DX	JNE SUM
MOV I,1	INC PUT
	XOR BX,BX
FOR1:	ADD BX,PUT

JMP FOR	int a,b,c,n;
SUM:	while(scanf("%d",&n)==1){
ADD COUNT,BX	a = sum(n);
INC PUT	b = sum(a);
XOR BX,BX	c = sum(b);
ADD BX,PUT	printf("%d\n",c);}
JMP FOR	return 0;
PRINTF:	}
XOR AX,AX	int sum(int n)
MOV AX,COUNT	{
CALL PRINT_NUM	int sum=0,digit;
DEC N	while(n!=0)
JMP FOR1	{
EXIT:	digit =n%10;//123%10=3
	sum= sum+digit;
	n=n/10;//123/10=12
MAIN ENDP	}
DEFINE_SCAN_NUM	return sum;
DEFINE_PRINT_NUM	}
DEFINE_PRINT_NUM_UN\$	;uva prblm 11332
END MAIN	
//uva prblm 11332	.MODEL SMALL
#include<stdio.h>	.STACK 100H
int sum(int n);	.DATA
int main()	DIGIT DW 0
{	N1 DW 0

N DW 0	MOV AX,C
N3 DW 0	
A DW 0	MOV N,0
B DW 0	MOV N,AX
C DW 0	XOR AX,AX
Z DW 3	
	CALL SUM
.CODE	MOV AX,C ;B=SUM(A)
MAIN PROC	
MOV AX,@DATA ;DATA SEGMENT INITIALIZE	MOV N,0
MOV DS,AX	MOV N,AX
	XOR AX,AX
INCLUDE 'EMU8086.INC'	
	CALL SUM
	MOV AX,C ;C=SUM(B)
XOR AX,AX	
XOR BX,BX ;CLEAR REGISTER	
XOR CX,CX	CALL PRINT_NUM
XOR DX,DX	PRINTN
CALL SCAN_NUM	
PRINTN	
MOV AX,CX	
MOV N,AX	MAIN ENDP
XOR AX,AX	
	SUM PROC
CALL SUM ;A=SUM(N)	



MOV A,10	//UVA PRBLM 11854
MOV C,0	#include<stdio.h>
	#include<math.h>
WHILE:	int main()
CMP N,0	{
JE EXIT	int a,b,c,d,i,n;
	while(scanf("%d%d%d",&a,&b,&c)==2)
MOV AX,N	{
DIV A	if(a==0&&b==0&&c==0)
ADD C,DX	break;
MOV N,AX	else
XOR AX,AX	{
XOR DX,DX	if(c==sqrt((a*a)+(b*b)))
JMP WHILE	printf("right\n");
EXIT:	else
RET	printf("wrong\n");
	}
	}
	return 0;
	}
	;UVA PRBLM 11854
DEFINE_SCAN_NUM	.MODEL SMALL
DEFINE_PRINT_NUM	.STACK 100H
DEFINE_PRINT_NUM_UN\$	.DATA
	A DW 0
	B DW 0
END MAIN	C DW 0

.CODE	MOV A,AX
MAIN PROC	CMP A,0 ;A==0 THEN EXIT
	JE EXIT
STRT:	XOR AX,AX
MOV AX,@DATA ;DATA SEGMENT INITIALIZE	
MOV DS,AX	;PRINTN "ENTER B"
	CALL SCAN_NUM
INCLUDE 'EMU8086.INC'	MOV AX,CX
	ADD B,AX
;FOR USER INPUT	IMUL B
	MOV BX,AX
DEFINE_SCAN_NUM	
DEFINE_PRINT_NUM	CMP BX,0
DEFINE_PRINT_NUM_UN\$	JE EXIT
	XOR AX,AX
	ADD BX,A
XOR CX,CX	
XOR AX,AX	
XOR BX,BX ;O ASIGN ALL THE RESISTER	
XOR DX,DX	;PRINTN "ENTER C"
	CALL SCAN_NUM
; PRINTN "ENTER A"	MOV AX,CX
	ADD C,AX
CALL SCAN_NUM	IMUL C
	MOV DX,AX
MOV AX,CX	
ADD A,AX	CMP DX,0
IMUL A	JE EXIT

XOR AX,AX	scanf("%d%d",&a,&b);
	if(a==0&&b==0)
	break;
CMP BX,DX ;C2==A2 *B2	while(a>0&&b>0)
JE PRINTR	{
PRINTN "WRONG"	
JMP EXIT	r1=a%10;
PRINTR:	r2=b%10;
PRINTN "RIGHT"	
	sum = r1+r2;
	a=a/2;
	b=b/2;
	if(sum+c>=10)
EXIT:	{
	c++;
MOV AH,4CH	}
INT 21H	
	}
END MAIN	if(c>0)
//uva 10035	{
#include<stdio.h>	printf("%d carry operation\n",c);
int main()	}
{	else
int a,b,r1,r2;	printf("No carry operation \n");
while(1)	}
{	return 0;
int sum=0,c=0;	}

;UVA PRBLM 10035	MOV D,10
	MOV E,2
.MODEL SMALL	
.STACK 100H	CALL SCAN_NUM
.DATA	;ADD P,CX
SUM DW 0	MOV A,CX
A DW 0	
B DW 0	PRINTN
R1 DW 0	
R2 DW 0	CALL SCAN_NUM
C dw ?	; ADD Q,CX
D DW 0	MOV B,CX
E DW 0	PRINTN
P DW 0	
.CODE	CMP A,0
MAIN PROC	JE EXIT
STRT:	CMP B,0
MOV AX,@DATA ;DATA SEGMENT INITIALIZE	JE EXIT
MOV DS,AX	
	WHILE:
INCLUDE 'EMU8086.INC'	CMP A,0
	JG CHECK
XOR CX,CX	CHECK:
XOR AX,AX	CMP B,0
XOR BX,BX ;O ASIGN ALL THE RESISTER	JG CAL
XOR DX,DX	JMP IF

CAL:

XOR AX,AX

ADD AX,A

DIV D

MOV R1,DX

XOR DX,DX ;R1

XOR AX,AX

ADD AX,B

DIV D

MOV R2,DX ;R2

XOR DX,DX

XOR AX,AX

MOV AX,R1 ;R2+R1

ADD AX,R2

MOV SUM,AX

XOR AX,AX

MOV AX,A

DIV E

MOV A,AX

XOR AX,AX ;A/2

XOR DX,DX

MOV AX,B

DIV E

MOV B,AX

XOR AX,AX

XOR DX,DX ;B/2

MOV AX,SUM

ADD AX,C

CMP AX,10

JGE L2

JMP WHILE

L2:

INC C

JMP WHILE

IF:

CMP C,0

JG PRINT1

PRINT1:

MOV AX,C

CALL PRINT\_NUM

PRINTN "CARRRY OPERATION"

JMP EXIT	
PRINT2:	while(n--)
PRINTN " NO CARRY OPERATION"	{
	char a[10];
	scanf("%s",a);
EXIT:	
	if(strlen(a)==5)
MOV AH,4CH	printf("3\n");
INT 21H	else
	{
	int c=0;
MAIN ENDP	if((a[0])=='o')
	c++;
	if((a[1])=='n')
	c++;
DEFINE_SCAN_NUM	if((a[2])=='e')
DEFINE_PRINT_NUM	c++;
DEFINE_PRINT_NUM_UN\$	if(c>=2)
END MAIN	printf("1\n");
	else
//uva 12289	printf("2\n");}}
#include<stdio.h>	return 0;
#include<string.h>	}
int main()	
{	;
int n;	#UVA PRBLM 1124
scanf("%d",&n);	
	.MODEL SMALL

.STACK 100H

.DATA

char db 30 DUP(?)

;ARRAY1 DB 100 DUB(?)

A DB 0

COUNT DB 0

COUNT1 DB 0

.CODE

MAIN PROC

STRT:

MOV AX,@DATA

MOV DS,AX ;DATA INITIALIZATION

INCLUDE 'EMU8086.INC'

;FOR USER INPUT

XOR CX,CX

XOR BX,BX ;CLEAR REGISTER

XOR AX,AX

XOR DX,DX

MOV BX,0

MOV COUNT,0

Input: ;get the string

mov ah, 1 ;input

int 21h

cmp al, 13

je after ;if enter then exit

mov char[bx], al

inc bx

INC COUNT

PRINTN

jmp Input ;jump to input

after:

CMP COUNT,5

JE PRINT3

XOR BX,BX

MOV BX,0

MOV AL,CHAR[BX]

CMP AL,'O'

JE L2

INC BX

MOV AL,0

MOV AL,CHAR[BX]

CMP AL,'N'

JE L2

INC BX	
MOV AL,0	EXIT:
MOV AL,CHAR[BX]	MOV AH,4CH
CMP AL,'E'	INT 21H
JE L2	
JMP IF	MAIN ENDP
L2:	
INC COUNT1	DEFINE_SCAN_NUM
	DEFINE_PRINT_NUM
IF:	DEFINE_PRINT_NUM_UN
CMP COUNT1,2	END MAIN
JGE PRINT1	//uva 11984
JNGE PRINT2	#include<stdio.h>
JMP EXIT	int main ()
	{
PRINT1:	int testCase; scanf ("%d", &testCase);
PRINTN "1"	int cases = 0;
JMP EXIT	
	while ( testCase-- ) {
PRINT2:	int c, d; scanf ("%d %d", &c, &d);
PRINTN "2"	
JMP EXIT	double f = (9.0/5.0) * c + 32;
	f += d;
PRINT3:	f -= 32;
PRINTN "3"	f *= 5;
JMP EXIT	f /= 9;
	printf ("Case %d: %.2lf\n", ++cases, f);
	}



```

    return 0;
}
;UVA PRBLM 11984

.MODEL SMALL
.STACK 100H
.DATA
TESTCASE DW 0
CASE DW 0
C DW 0
D DW 0
F DW 0
M DW 0
N DW 0
P DW 0
Q DW 0
.CODE
MAIN PROC
    STRT:
    MOV AX,@DATA
    MOV DS,AX    ;DATA INITIALIZATION

    INCLUDE 'EMU8086.INC'
;FOR USER INPUT

```

```

XOR CX,CX
XOR BX,BX    ;CLEAR REGISTER
XOR AX,AX
XOR DX,DX
MOV M,9
MOV N,5
MOV F,32

CALL SCAN_NUM
MOV TESTCASE,CX
PRINTN

WHILE:
    CMP TESTCASE,0
    JE STRT

    CALL SCAN_NUM
    MOV C,CX
    PRINTN

    CALL SCAN_NUM
    MOV D,CX
    PRINTN

    MOV AX,M
    DIV N

```

MOV P,AX

XOR AX,AX

XOR DX,DX

MOV AX,P

MUL C

MOV BX,AX

ADD BX,F

ADD BX,D

SUB BX,F

MOV AX,BX

MUL N

XOR DX,DX

DIV M

PRINT "CASE: "

CALL PRINT\_NUM

PRINTN

XOR DX,DX

XOR AX,AX

JMP STRT

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN\$

END MAIN

#include<stdio.h>

int main()

{

int a;

int n;

while(1)

{

scanf("%d",&a);

if(a==0)

break;

else

{

if(a%11==0)

printf("%d is a multiple of 11.\n",a);

else

printf("%d is not a multiple of 11.\n",a);

}

```

    }
    return 0;
}
;UVA PRBLM 10970

```

```

.MODEL SMALL

```

```

.STACK 100H

```

```

.DATA

```

```

M DW 0

```

```

N DW 0

```

```

A DW 0

```

```

.CODE

```

```

MAIN PROC

```

```

    STRT:

```

```

    MOV AX,@DATA ;DATA SEGMENT INITIALIZE

```

```

    MOV DS,AX

```

```

    INCLUDE 'EMU8086.INC'

```

```

;FOR USER INPUT

```

```

    DEFINE_SCAN_NUM

```

```

    DEFINE_PRINT_NUM

```

```

    DEFINE_PRINT_NUM_UN$

```

```

    XOR CX,CX

```

```

    XOR AX,AX

```

```

    XOR BX,BX ;O ASIGN ALL THE RESISTER

```

```

    XOR DX,DX

```

```

    MOV N,11

```

```

    CALL SCAN_NUM

```

```

    ADD A,CX

```

```

    MOV M,CX

```

```

    PRINTN

```

```

    CMP M,0

```

```

    JE EXIT

```

```

    JMP CALCULATION

```

```

    CALCULATION:

```

```

        MOV AX,M

```

```

        DIV N

```

```

        CMP DX,0

```

```

        JE PRINT1

```

```

        JNE PRINT2

```

```

    PRINT1:

```

```

        XOR DX,DX

```

```

        XOR AX,AX

```

```

        MOV AX,A

```

```

        CALL PRINT_NUM

```

```

        PRINTN "IS A MULTIPLE OF 11."

```

```

JMP STRT

PRINT2:
XOR DX,DX
XOR AX,AX
    MOV AX,A
    CALL PRINT_NUM
    PRINTN " :IS NOT A MULTIPLE OF 11."

JMP STRT

EXIT:

    MOV AH,4CH
    INT 21H

;
END MAIN

```

```

//UVA PRBLM 10079

#include<stdio.h>

int main()
{
    int p;

    long long int n;

    while(scanf("%lld",&n))
    {
        if(n<0)
            break;

        printf("%lld\n",1+ (n*(n+1)/2));
    }

    return 0;
}

;UVA PRBLM 10079

.MODEL SMALL

.STACK 100H

.DATA
A DW 0
N DW 0
B DW ?
R DW ?
K DW ?

.CODE

MAIN PROC

    STRT:

    MOV AX,@DATA

```

MOV DS,AX ;DATA INITIALIZATION

ADD AX,1 ;+1

INCLUDE 'EMU8086.INC'

PRINTN " "

;FOR USER INPUT

CALL PRINT\_NUM

JMP STRT

XOR CX,CX

XOR BX,BX ;CLEAR REGISTER

XOR AX,AX

XOR DX,DX

EXIT:

MOV BX,2

MOV AH,4CH

INT 21H

CALL SCAN\_NUM ;SCAN N

PRINT

MAIN ENDP

MOV AX,CX

DEFINE\_SCAN\_NUM

ADD A,AX

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN

CMP AX,0

END MAIN

JE EXIT ;N<0 BREAK

//UVA PRBLM 11498

#include<stdio.h>

ADD AX,1 ;ELSE N+1

int main()

{

MUL A ;N\*N+1

int n,m,i,x,y,k;

while(scanf("%d",&k)==1)

DIV BX ;/2

{

if(k==0)	.MODEL SMALL
break;	.STACK 100H
scanf("%d%d",&n,&m);	.DATA
for(i=0; i<k; i++)	A DW 0
{	N DW 0
scanf("%d%d",&x,&y);	M DW 0
if(n==x    m==y)	I DW 0
printf("divisa\n");	K DW 0
else if(x>n && y>m)	X DW 0
printf("NE\n");	Y DW 0
else if(x<n&&y>m)	.CODE
{	MAIN PROC
printf("NO\n");	STRT:
}	MOV AX,@DATA
else if(x<n&&y<m)	MOV DS,AX   ;DATA INITIALIZATION
{	
printf("SO\n");	
}	INCLUDE 'EMU8086.INC'
else if(x>n&&y<m)	;FOR USER INPUT
{	
printf("SE\n");	
}	
}	XOR CX,CX
}	XOR BX,BX   ;CLEAR REGISTER
return 0;	XOR AX,AX
}	XOR DX,DX
;#UVA PRBLM 11498	

```
CALL SCAN_NUM
MOV K,CX
PRINTN    ;CX = K
```

```
CMP K,0
JE EXIT
```

```
CALL SCAN_NUM
MOV N,CX
PRINTN
```

```
CALL SCAN_NUM
MOV M,CX
PRINTN
```

```
MOV DX,0 ;I=0=DX
```

```
LOOP1:
    CMP DX,K
    JG EXIT
```

```
CALL SCAN_NUM
MOV AX,CX    ;X=AX
PRINTN
```

```
CALL SCAN_NUM
MOV BX,CX    ;Y=BX
```

```
CMP AX,N
```

```
JE CHECK1 ;X== N
```

```
CMP AX,N
JG CHECK2 ;X>N
```

```
CMP AX,N ;X<N
JL CHECK3
```

```
CMP AX,N
JL CHECK4 ;X<N
```

```
CHECK1:
    CMP BX,M ;Y==M
    JE PRINT1
```

```
CHECK2:
    CMP BX,M
    JG PRINT2 ;Y>M
```

```
CHECK3:
    CMP BX,M
    JG PRINT3
```

```
CHECK4:
    CMP BX,M
    JL PRINT4
```

```
PRINT1:
    PRINTN "DIVISA"
    INC DX
```

JMP LOOP1	DEFINE_SCAN_NUM
PRINT2:	DEFINE_PRINT_NUM
PRINTN "NE"	DEFINE_PRINT_NUM_UN\$
INC DX	END MAIN
JMP LOOP1	//UVA PRBLM 12646
	#include <stdio.h>
PRINT3:	int main()
PRINTN "NO"	{
INC DX	int a,b,c;
JMP LOOP1	while(scanf("%d %d %d", &a, &b, &c) == 3)
	{
PRINT4:	if(a == b && b == c)
PRINTN "SO"	printf("*\n");
INC DX	else if(a != b && b == c)
	printf("A\n");
	else if(a != b && a == c)
	printf("B\n");
JMP LOOP1	else if(a == b && a != c)
	printf("C\n");
	}
	return 0;
	}
	;UVA PRBLM 12646
EXIT:	
MOV AH,4CH	.MODEL SMALL
INT 21H	.STACK 100H
	.DATA
MAIN ENDP	V DW 0



T DW 0

.CODE

MAIN PROC

MOV AX,@DATA ;DATA SEGMENT INITIALIZE

MOV DS,AX

INCLUDE 'EMU8086.INC'

;FOR USER INPUT

XOR CX,CX

XOR AX,AX

XOR BX,BX ;O ASIGN ALL THE RESISTER

XOR DX,DX

CALL SCAN\_NUM

;SCAN A

MOV AX,CX

PRINTN

; XOR CX,CX

CALL SCAN\_NUM

;SCAN B

MOV BX,CX

PRINTN

;XOR CX,CX

CALL SCAN\_NUM ;SCAN C

PRINTN

MOV DX,CX

LEVEL1:

CMP AX,BX

JE LEVEL3 ;AX==BX

JNE LEVEL8

LEVEL2:

CMP AX,DX ;AX==DX

JE LEVEL4

LEVEL3:

CMP BX,DX ;BX==DX

JE PRINT1

LEVEL4:

CMP AX,BX

JNE PRINT3 ;AX!=BX

LEVEL5:

CMP AX,DX ;AX!=DX

JNE LEVEL7

LEVEL6:

CMP BX,DX	MOV AH,4CH
JE PRINT2 ;BX==DX	INT 21H
LEVEL7:	
CMP AX,BX	
JE PRINT4 ;AX==BX	MAIN ENDP
LEVEL8:	
CMP AX,DX ;AX==DX	DEFINE_SCAN_NUM
JE PRINT3	DEFINE_PRINT_NUM
	DEFINE_PRINT_NUM_UN
PRINT1:	
PRINTN "*"	
JMP EXIT	;
	END MAIN
PRINT2:	//uva 12917
PRINTN "A"	#include<stdio.h>
JMP EXIT	int main()
	{
PRINT3:	int x,y,z;
PRINTN "B"	while(scanf("%d%d%d",&x,&y,&z)==3)
JMP EXIT	{
PRINT4:	if(x<=(z-y))
PRINTN "C"	printf("Props win!\n");
JMP EXIT	else
	printf("Hunters win!\n");
	}
	return 0;
EXIT:	}
	;
	#UVA PRBLM 12917

.MODEL SMALL

.STACK 100H

.DATA

A DW 0

N DW 0

B DW ?

R DW ?

K DW ?

.CODE

MAIN PROC

STRT:

MOV AX,@DATA

MOV DS,AX ;DATA INITIALIZATION

INCLUDE 'EMU8086.INC'

;FOR USER INPUT

XOR CX,CX

XOR BX,BX ;CLEAR REGISTER

XOR AX,AX

XOR DX,DX

CALL SCAN\_NUM

MOV DX,CX ;X INPUT

PRINTN

CALL SCAN\_NUM

MOV BX,CX ;Y INPUT

PRINTN

CALL SCAN\_NUM

MOV AX,CX ;Z INPUT

PRINTN

SUB AX,BX

MOV A,AX ;Z-Y

CMP DX,A

JLE PRINT12

PRINTN "Hunters win!"

jmp strt

PRINT12:

printn "Props win!"

jmp strt

EXIT:

MOV AH,4CH

INT 21H

```

MAIN ENDP

DEFINE_SCAN_NUM
    DEFINE_PRINT_NUM
    DEFINE_PRINT_NUM_UN
END MAIN

//uva 12952
#include<stdio.h>

int main()
{
    int a,b;
    while( scanf("%d%d",&a,&b)!=EOF)
    {
        if(a>=b)
            printf("%d\n",a);
        else
            printf("%d\n",b);
    }
    return 0;

}

;#UVA PRBLM 12952

.MODEL SMALL
.STACK 100H
.DATA
A DW 0
N DW 0
B DW ?

```

```

R DW ?
K DW ?

.CODE
MAIN PROC
    STRT:
    MOV AX,@DATA
    MOV DS,AX    ;DATA INITIALIZATION

    INCLUDE 'EMU8086.INC'
    ;FOR USER INPUT

    XOR CX,CX
    XOR BX,BX    ;CLEAR REGISTER
    XOR AX,AX
    XOR DX,DX

    CALL SCAN_NUM ;DX = A
    MOV DX,CX
    PRINTN

    CALL SCAN_NUM
    MOV BX,CX    ;BX = B
    PRINTN

```

CMP DX,BX	while(n--)
JGE L1	{
L2:	scanf("%d",&m);
MOV AX,BX	printf("Case #%%d: %d\n",count++,2*m-1);
CALL PRINT_NUM	}
JMP EXIT	return 0;
L1:	}
MOV AX,DX	;#UVA PRBLM 12992
CALL PRINT_NUM	
	.MODEL SMALL
	.STACK 100H
	.DATA
EXIT:	A DW 0
MOV AH,4CH	N DW 0
INT 21H	B DW 0
	I DW 0
	J DW 0
MAIN ENDP	.CODE
	MAIN PROC
DEFINE_SCAN_NUM	STRT:
DEFINE_PRINT_NUM	MOV AX,@DATA
DEFINE_PRINT_NUM_UN\$	MOV DS,AX ;DATA INITIALIZATION
END MAIN	
//UVA PRBLM 12992	
#include<stdio.h>	
int main()	INCLUDE 'EMU8086.INC'
{	;FOR USER INPUT
int n,m,count = 1;	
scanf("%d",&n);	

```

XOR CX,CX
XOR BX,BX ;CLEAR REGISTER
XOR AX,AX
XOR DX,DX
MOV B,2
CALL SCAN_NUM
MOV BX,CX
mov a,0
PRINTN

```

WHILE:

```

CMP BX,A
JE EXIT

```

```

CALL SCAN_NUM
MOV AX,CX
PRINTN

```

```

MUL B
SUB AX,1
MOV DX,AX
INC A

```

```

PRINT "Case #"
MOV AX,A
CALL PRINT_NUM
PRINT ":"

```

```

MOV AX,DX
CALL PRINT_NUM
PRINTN

JMP WHILE

```

EXIT:

```

MOV AH,4CH
INT 21H

```

MAIN ENDP

```

DEFINE_SCAN_NUM
DEFINE_PRINT_NUM
DEFINE_PRINT_NUM_UN

```

END MAIN

//UVA PRBLM 13012

#include<stdio.h>

int main()

{

int n,m,i,count;

while(scanf("%d",&n)!=EOF)

{	MOV AX,@DATA
count =0;	MOV DS,AX   ;DATA INITIALIZATION
for(i=0;i<5;i++){	INCLUDE 'EMU8086.INC'
scanf("%d",&m);	;FOR USER INPUT
if(n==m)	
count++;	
}	
	XOR CX,CX
	XOR BX,BX   ;CLEAR REGISTER
printf("%d\n",count);	XOR AX,AX
}	XOR DX,DX
return 0;	
}	call scan_num
;#UVA PRBLM 13012	MOV BX,CX
	PRINTN
	MOV A,0
.MODEL SMALL	
.STACK 100H	
.DATA	
A DW 0	CAL:
N DW 0	CMP BX,A
B DW 0	JE PRINT1
I DW 0	INC A
J DW 0	CALL SCAN_NUM
.CODE	MOV DX,CX
MAIN PROC	PRINTN
STRT:	

CMP DX,BX

JE LOOP1

JMP CAL

LOOP1:

INC J

JMP CAL

PRINT1:

MOV AX,J

;PRINTN

CALL PRINT\_NUM

PRINTN

EXIT:

MOV AH,4CH

INT 21H

MAIN ENDP

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UN\$

END MAIN





