**1.UVA\_10055\_USING\_ASSEMBLY**

include 'emu8086.inc'

.MODEL SMALL

.STACK 100H

.CODE ; main code section

MAIN PROC ; main procedure

start:

MOV AH, 1 ; first input from user

INT 21H

MOV CL, AL ; move data AL into CL

MOV DL, ' ' ; Display a single space

MOV AH, 2

INT 21H

MOV AH, 1 ; second input from user

INT 21H

MOV BL, AL ; move data AL into BL

PRINTN ; displaying newline

CMP CL, BL ; comparing CL and BL

JG first\_grater\_than\_second

SUB BL, CL ; BL=BL-CL

ADD BL, 48 ; BL-=48

MOV DL, BL ; set BL into DL for

; displaying

MOV AH, 2

INT 21H

PRINTN

JMP start

; if first > second

first\_grater\_than\_second:

SUB CL, BL

ADD CL, 48 ; CL+=48

MOV DL, CL ; move CL into DL

MOV AH, 2

INT 21H

PRINTN

JMP start

EXIT:

MOV AH, 4CH

INT 21H

MAIN ENDP

END MAIN

**1.UVA\_10055 \_JAVA**

package Hasmot;

import java.util.Scanner;

public class Main

{

public static Scanner m = new Scanner(System.in);

public static void main(String[] args)

{

while (m.hasNext())

{

long a = m.nextLong();

long b = m.nextLong();

if (a > b)

{

System.out.println(a - b);

}

else {

System.out.println(b - a);

}

}

}

}

**2.UVA\_10783 \_ASSEMBLY**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

sum DW ? ; sum=0

t DB ? ; t=0

t2 DB ? ; t2=0

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

;scanf("%d", &t)

MOV AH, 1 ; AH=1

INT 21H

MOV t, AL ; t=AL

SUB t, 48 ; t=t-48

MOV t2, 1 ; t2=1

TEST\_CASE:

PRINTN ; print newline

MOV AL, t ; AL=t

CMP t2, AL ; compare t2 and t

JG EXIT ; jump if greater

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

MOV AH, 1 ; AH=1

INT 21H

MOV BL, AL ; BL=AL

SUB BL, 48 ; BL-=48

CBW ; convert byte to word

PRINT " " ; print single space

MOV AH, 1 ; AH=1

INT 21H

SUB AL, 48 ; AL-=48

CBW ; convert byte to word

MOV CX, AX ;CX=AX

MOV sum,0 ; sum = 0

;for(j=a; j<=b; j++)

FOR\_START:

CMP BX,CX ; compare BX and CX

JG EXIT\_FOR ; jump if greater

MOV AX,BX ; AX=BX

MOV DL,2 ; DL=2

DIV DL ; AX/DL

CMP AH,0 ; AH=(AX%DL), compare

;AH and 0

JE IncreamentAndBackToforLoop:

ADD SUM, BX ; sum+=BX

IncreamentAndBackToforLoop:

INC BX ; BX++

JMP FOR\_START ; jump to FOR\_START

EXIT\_FOR:

PRINTN ; print newline

PRINT "CASE " ; displaying "CASE "

MOV AH, 2 ; AH=2

MOV DL, t2 ; DL=t2

ADD DL, 48 ; DL+=48

INT 21H

PRINT " :" ; displaying " :"

MOV AH, 2 ; AH=2

MOV DX, sum ; DX=sum

ADD DL, 48 ; DL+=48

INT 21H

INC t2 ; t2++

JMP TEST\_CASE

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main prc end

END MAIN

**2.UVA\_10783\_C**

#include<stdio.h>

int main()

{

int i, j, sum, a, b, t;

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

{

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

{

sum=0;

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

for(j=a; j<=b; j++)

{

if((j%2)!=0)

sum=sum+j;

}

printf("Case %d: %d\n", i, sum);

}

}

return 0;

}

**3.UVA\_10071 \_ASSEMBLY**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

u db ?

t db ?

.code

main proc

mov ax, @data

mov ds, ax

;while(scanf("%d%d", &u, &t)==2)

for:

mov s, 0 ; s=0

mov ah, 1 ; input u

int 21h

mov u, al

sub u, 48 ; u-=48

print ' '

mov ah, 1 ; input t

int 21h

printn ; displaying newline

mov t, al

sub t, 48 ; t-=48

mov al, 2 ; al=2

mul u ; ax = al\*u

xor ah, ah ; ah=0

mul t ; ax=al\*t

xor ah, ah ; ah=0

mov ah, 2

mov dl, al ; result

add dl, 48 ; dl+=48

int 21h

jmp for

mov ah, 4ch

int 21h

main endp

end main

**3.UVA\_10071\_C**

#include<stdio.h>

int main()

{

int u, t, s;

while(scanf("%d%d", &u, &t)==2)

{

s=u\*2\*t;

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

}

return 0;

}

**4.UVA\_10346 \_ASSEMBLY**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

n db ?

i db ?

k db ?

.code

main proc

mov ax, @data ; import data

mov ds, ax

; while(scanf("%d%d",&N,

; &k)==1 && k>1)

while\_start:

mov ah, 1 ; input N

int 21h

sub al, 48 ; al-=48

mov n, al ; N=al

mov i, al

print ' ' ; print single space

mov ah, 1 ; input k

int 21h

sub al, 48 ; al-=48

mov k, al ; k=al

cmp al, 1 ;if k<=1

jle Exit ; then exit

printn

;while(n>=k)

while2:

mov al, n ; al=n

cmp al, k ; compare al and k

jl exit\_while2 ; if al<k then exit while2

mov al, n ; al=n

xor ah, ah ; ah=0

mov dl, k ; dl=k

div dl ; al=ax/dl

add i, al ; i+=al

mov n, al ; n=(n/k)+(n%k)

add n, ah

jmp while2

exit\_while2:

printn ; displaying newline

mov dl, i ; displaying result

add dl, 48

mov ah, 2

int 21h

printn

jmp while\_start ; jump to while\_start

Exit:

mov ah, 4ch

int 21h

main endp

end main

**4.UVA\_10346\_C**

#include<stdio.h>

int main()

{

int i,n,k;

while(scanf("%d %d",&n,&k)==2 && k>1)

{

i=n;

while(n>=k)

{

i= i+n/k;

n=(n/k)+(n%k);

}

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

}

return 0;

}

**5.UVA\_11150 \_ASSEMBLY**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

N db ?

temp db ?

e db ?

.code

main proc

mov ax, @data ; import data

mov ds, ax

;while(scanf("%d",&N)==1)

while\_start:

mov ah, 1 ; input N

int 21h

sub al, 48 ; al-=48

mov N, al ; N=al

mov temp, al ; temp=N+1

add temp, 1

mov e, al ; e=N

printn

;while(temp>=3)

while2:

mov al, temp ; al=temp

xor ah, ah ; ah=0

mov dl, 3

div dl

add e, al ; e+=(temp/3)

xor dl, dl ; dl=0

add dl, ah ; temp=(temp%3)+(temp/3)

add dl, al

mov al, dl

mov temp, dl

cmp al, 3 ; if temp>=3

jge while2 ; jump while2

mov ah, 2

mov dl, e ; displaying result

add dl, 48

int 21h

printn ; displaying newline

jmp while\_start ; jump to while\_start

Exit:

mov ah, 4ch

int 21h

main endp

end main

**5.UVA\_11150 \_C**

#include<stdio.h>

int main()

{

int N;

int temp;

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

{

temp=N+1;

int e=N;

while(temp>=3)

{

e=e+(temp/3);

temp=(temp%3)+(temp/3);

}

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

}

return 0;

}

**6.UVA\_11172 \_ASSEMBLY**

INCLUE 'emu8086.inc'

.MODEL SMALL

.STACK 100h

.DATA ; data section

.CODE ; code section

MAIN PROC ; main procedure start

MOV AH, 1 ; input for test

; case from user

INT 21H

MOV CL, AL ; CL = AL

SUB CL, 48 ; CL-=48

PRINTN ; displaying newline

start\_loop:

; first input from user

; for each test case

MOV AH, 1

INT 21h

MOV BL, AL

PRINT " " ; displying a single space

MOV AH, 1 ; second input from user

; for each test case

INT 21H

PRINTN ; displaying newline

CMP BL, AL ; comparing bl and al

JG greater ; jump if greater than

JE equal ; jump if equal

; else

PRINT "<" ; displaying '<' with newline

JMP last\_testCase\_check

greater:

PRINT ">" ; displaying '>' with newline

JMP last\_testCase\_check

equal:

PRINTN "=" ; displaying '=' with newline

JMP last\_testCase\_check

last\_testCase\_check:

DEC cl

JZ exit

JMP start\_loop

exit:

MOV AH, 4CH

INT 21H

MAIN ENDP

END MAIN

**6.UVA\_11172 \_C**

#include<stdio.h>

int main()

{

int t, a, b, i;

scanf("%d", &t);

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

{

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

if(a>b)

printf(">\n");

else if(a<b)

printf("<\n");

else

printf("=\n");

}

return 0;

}

**7.UVA\_10970 \_ASSEMBLY**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

totalCut dw ?

M dw ?

N dw ?

.code ; code section

main proc

mov ax, @data

mov ds, ax

call SCAN\_NUM ; procedure for input

mov M, cx

printn

call SCAN\_NUM ; procedure for input

mov N, cx

printn

mov ax, M

mul N

mov cx, ax

sub cx, 1

mov ax, cx ; procedure for output

call PRINT\_NUM\_UNS

mov ah, 4ch

int 21h

main endp

; define for input

; and output

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM\_UNS

end main

**7.UVA\_10970 \_JAVA**

import java.util.Scanner;

public class Main {

public static Scanner s=new Scanner(System.in);

public static void main(String[] args) {

long M, N, totalCut;

while(s.hasNext()){

M=s.nextLong();

N=s.nextLong();

totalCut=(M\*N)-1;

System.out.println(totalCut);

}

}

}

**8.UVA\_11044 \_ASSEMBLY**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

first db ?

second db ?

t db ?

.code ; code section

main proc

mov ax, @data ; import data

mov ds, ax

mov ah, 1 ; input t

int 21h

sub al, 48

mov t, al

printn

while:

xor ax, ax

xor cx, cx

xor dx, dx

mov ah, 1 ; first input

int 21h

sub al, 48 ; al-=48

xor ah, ah ; ah = 0

mov dl, 3 ; dl=3

div dl ; al = ax/dl

mov first, al ; first=al

print " " ; print single space

mov ah, 1 ; second input

int 21h

sub al, 48 ; al-=48

xor ah, ah ; ah=0

mov dl, 3 ; dl=3

div dl ; al=ax/dl

mov second, al ; second=al

printn ; print newline

mov al, first

mov dl, second

mul dl ; ax=al\*cl

mov ah, 2 ; output

mov dl, al ; dl=al

add dl, 48 ; dl+=48

int 21h

dec t

mov al, t

dec al

cmp al, 0

jl Exit

printn

jmp while

Exit:

mov ah, 4ch

int 21h

main endp

end main

**8.UVA\_11044 \_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int t,n,m;

cin>>t;

while(t--)

{

cin>>n>>m;

cout<<(n/3)\*(m/3)<<endl;

}

return 0;

}

**9.UVA\_11364\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

testCase dw 0 ; testCase=0

noOfStores dw 0 ; noOfStores=0

max dw 0 ; max=0

min dw 99 ; min=99

x dw 0 ; x=0

.code ; code section

main proc ; main proc

mov ax, @data ; data import

mov ds, ax

mov ah, 1 ; cin>>testCase;

int 21h

sub al, 48

xor ah, ah

mov testCase, ax

printn

while1: ; while(testCase--)

mov ah, 1

int 21h

sub al, 48

xor ah, ah

mov noOfStores, ax

printn

while2: ;while(noOfStores--)

call SCAN\_NUM ; cin>>x;

mov x, cx

printn

mov ax, max

cmp ax, x

jl max\_lebel ; if(max<x)

mov ax, min

cmp ax, x

jg min\_lebel ; if(min>x)

max\_lebel:

MOV ax, x

mov max, ax

jmp after

min\_lebel:

mov ax, x ; max=x;

mov min, ax

after:

dec noOfStores

cmp noOfStores, 0

je Exit\_while2

jmp while2

Exit\_while2:

mov ax, max

sub ax, min

mov dl, 2

mul dl

; cout<<(max-min)\*2<<endl;

call PRINT\_NUM\_UNS

dec testCase

cmp testCase, 0

je Exit

printn

jmp while1

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

end main

**9.UVA\_11364\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int testCase, noOfStores;

int x;

cin>>testCase;

while(testCase--)

{

int max=0, min=99;

cin>>noOfStores;

while(noOfStores--)

{

cin>>x;

if(max<x)

max=x;

if(min>x)

min=x;

}

cout<<(max-min)\*2<<endl;

}

return 0;

}

**10. UVA\_11777\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

Term1 dw ?

Term2 dw ?

Final dw ?

Attendance dw ?

Class\_Test1 dw ?

Class\_Test2 dw ?

Class\_Test3 dw ?

T dw ?

m1 dw ?

AvgClassTest dw ?

totalMarks dw ?

i dw ?

.code ; code section

main proc

mov ax, @data ; import data

mov ds, ax

while\_start: ; while(scanf("%d", &T)==1)

call SCAN\_NUM

mov t, cx

printn

mov i, 1

for: ; for(i=1; i<=T; i++)

mov m1, 0

mov AvgClassTest, 0

mov totalMarks, 0

;scanf("%d%d%d%d%d%d%d",

&Term1, &Term2, &Final,

&Attendance, &Class\_Test1,

&Class\_Test2, &Class\_Test3);

xor cx, cx ; cx=0

CALL SCAN\_NUM

mov Term1, cx ; Term1=cx

printn

xor cx, cx

CALL SCAN\_NUM ; Term2 Input

mov Term2, cx

printn

xor cx, cx

CALL SCAN\_NUM ; Final Input

mov Final, cx

printn

xor cx, cx

CALL SCAN\_NUM

mov Attendance, cx ; Attendance

printn

xor cx, cx

CALL SCAN\_NUM ; Class\_Test1

mov Class\_Test1, cx

printn

xor cx, cx

CALL SCAN\_NUM ; Class\_Test2

mov Class\_Test2, cx

printn

xor cx, cx

CALL SCAN\_NUM ; Class\_Test3

mov Class\_Test3, cx

printn

; find minimum ClassTest Mark

mov ax, Class\_Test1

cmp ax, Class\_Test2

jg greater\_than2

cmp ax, Class\_Test3

jg minimum\_C3

mov ax, Class\_Test1

mov m1, ax

jmp check\_end

greater\_than2:

mov ax, Class\_Test2

cmp ax, Class\_Test3

jle minimum\_c2

mov ax, Class\_Test3

mov m1, ax

jmp check\_end

minimum\_C3:

mov ax, Class\_Test3

mov m1, ax

jmp check\_end

minimum\_c2:

mov ax, Class\_Test2

mov m1, ax

check\_end: ; AvgClassTest =

((Class\_Test1+Class\_

Test2+Class\_Test3) - m2)/2;

mov ax, Class\_Test1

add ax, Class\_Test2

add ax, Class\_Test3

sub ax, m1

mov dl, 2

div dl

xor ah, ah

mov AvgClassTest, ax

mov ax, Term1

add ax, Term2

add ax, Attendance

add ax, Final

add ax, AvgClassTest

mov totalMarks, ax

;totalMarks = Term1+Term2+Attendance

+Final+AvgClassTest;

mov ax, totalMarks

; check grade

cmp ax, 60

jl lessThan60

cmp ax, 70

jl lessThan70

cmp ax, 80

jl lessThan80

cmp ax, 90

jl lessThan90

cmp ax, 100

jle lessOrEqual100

jmp check

lessOrEqual100:

print "Case "

mov ax, i

CALL PRINT\_NUM\_UNS

printn ": A"

jmp check

lessThan90:

print "Case "

mov ax, i

CALL PRINT\_NUM\_UNS

printn ": B"

jmp check

lessThan80:

print "Case "

mov ax, i

CALL PRINT\_NUM\_UNS

printn ": C"

jmp check

lessThan70:

print "Case "

mov ax, i

CALL PRINT\_NUM\_UNS

printn ": D"

jmp check

lessThan60:

print "Case "

mov ax, i

CALL PRINT\_NUM\_UNS

printn ": F"

check:

inc i

mov ax, i

cmp ax, T

jg exit\_for

jmp for

exit\_for:

jmp while\_start

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM\_UNS

end main

**10.UVA\_11777\_C**

#include <stdio.h>

#include <stdlib.h>

int main()

{

int Term1, Term2, Final, Attendance, Class\_Test1, Class\_Test2, Class\_Test3, T, m1, m2;

double AvgClassTest, totalMarks;

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

{

int i;

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

{

m1=0;

m2=0;

AvgClassTest=0;

totalMarks=0;

scanf("%d%d%d%d%d%d%d", &Term1, &Term2, &Final, &Attendance, &Class\_Test1,

&Class\_Test2, &Class\_Test3);

m1 = (Class\_Test1<=Class\_Test2)? Class\_Test1:Class\_Test2;

m2 = (m1<=Class\_Test3)? m1:Class\_Test3;

AvgClassTest = ((Class\_Test1+Class\_Test2+Class\_Test3) - m2)/2;

totalMarks = Term1+Term2+Attendance+Final+AvgClassTest;

if(totalMarks>=90 && totalMarks<=100)

{

printf("Case %d: A\n", i);

}

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

{

printf("Case %d: B\n", i);

}

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

{

printf("Case %d: C\n", i);

}

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

{

printf("Case %d: D\n", i);

}

else if(totalMarks<60)

{

printf("Case %d: F\n", i);

}

}

}

return 0;

}

**11. UVA\_12279\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

N dw 0

j dw 0

gt dw 0

st dw 0

i dw 0

p dw 0

.code ; code section

main proc

mov ax, @data

mov ds, ax

mov j, 1

while1: ; while(cin>>N)

CALL SCAN\_NUM

mov N, cx

printn

cmp cx, 0

je Exit

mov gt, 0

mov st, 0

mov i, 0

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

For\_start:

CALL SCAN\_NUM

printn ; print newline

cmp cx, 0

je gt\_Inc

inc st ; st++

jmp after

gt\_Inc:

inc gt ; gt++

after:

inc i ; i++

mov ax, i

cmp ax, N

jge exit\_for

jmp For\_start

exit\_for:

; output

print "Case "

mov ax, j

CALL PRINT\_NUM\_UNS

inc j ; j++

print ": "

mov ax, st

sub ax, gt ; ax=st-gt

CALL PRINT\_NUM\_UNS

jmp while1

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

end main

**11.UVA\_12279\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int N, p;

int j=0;

while(cin>>N){

if(N==0)

break;

int gt=0, st=0;

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

{

cin>>p;

if(p==0)

gt++;

else

st++;

}

cout<<"Case "<<++j<<": "<<st-gt<<endl;

}

return 0;

}

**12. UVA\_136\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

.CODE ; code section

MAIN PROC ; main proc start

printn "The 1500'th ugly number is 859963392"

Exit:

MOV AH, 4CH

INT 21H

MAIN ENDP ; main proc end

END MAIN

**12. UVA\_136\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

cout<<"The 1500'th ugly number is 859963392"<<endl;

return 0;

}

**13. UVA\_13025\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

.CODE ; code section

MAIN PROC ; main proc start

; diplaying result

printn "May 29, 2013 Wednesday"

Exit:

MOV AH, 4CH

INT 21H

MAIN ENDP ; main proc end

END MAIN

**13. UVA\_13025\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

cout<<"May 29, 2013 Wednesday"<<endl;

return 0;

}

**14. UVA\_573\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

ini dw ?

dayCount dw ?

H dw ?

U dw ?

D dw ?

F dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ;input H

mov H, cx

printn

call scan\_num ;input U

mov U, cx

printn

call scan\_num ;input D

mov D, cx

printn

call scan\_num ;input F

mov F, cx

printn

mov ax, H ; if H==0 then exit

cmp ax, 0

je Exit

;F=U\*F/100

mov ax, U

mov cx, F

mul cx

mov cx, 100

div cx

mov F, ax

mov dayCount, 0 ;daycount=0

mov ini, 0 ;ini=0

while2: ;while(1)

inc dayCount ;daycount++

mov ax, U

jle if2 ;if(U>0)

mov ax, ini

add ax, U ;initial+=U

mov ini, ax

if2: ;if(initial>H)

mov ax, U

sub ax, F

mov U, ax ;U-=F

mov ax, ini

cmp ax, H

jle after

; result

print "success on day "

mov ax, dayCount

jmp exit\_while2

after: ; ini-=D

mov ax, ini

sub ax, D

mov ini, ax

cmp ax, 0 ;if(ini<0)

jge after2

;result2

print "failure on day "

mov ax, dayCount

jmp exit\_while2

after2:

jmp while2

exit\_while2:

jmp while

Exit:

MOV AH, 4CH

INT 21H

MAIN ENDP ; main proc end

; define function

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**14. UVA\_573\_C++**

#include<bits/stdc++.h>

using namespace std;

double initial=0;

int main()

{

double H, U, D, F;

int daycount;

while(cin>>H>>U>>D>>F && H)

{

F=U\*F/100;

daycount=0;

double initial=0;

while(1)

{

daycount++;

if(U>0)

initial+=U;

U-=F;

if(initial>H)

{

cout<<"success on day "<<daycount<<endl;

break;

}

initial-=D;

if(initial<0)

{

cout<<"failure on day "<<daycount<<endl;

break;

}

}

}

return 0;

}

**15. UVA\_694\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

A dw ?

L dw ?

count dw ?

temp dw ?

T dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while: ; while start

mov T, 0

call scan\_num ;input A

mov A, cx

printn

call scan\_num ;input L

mov L, cx

printn

mov ax, A

cmp ax, 0

je check\_L

jmp after

check\_L: ; checking L

mov ax, L

cmp ax, 0

je Exit

after:

mov ax, A

mov temp, ax

mov count, 0

while2: ;while2 start

mov ax, A

cmp ax, L

jg exit\_while2

mov ax, A

cmp ax, 1 ;cmp ax and 1

je break\_count

xor dx, dx

mov ax, A

mov cx, 2

div cx

cmp dx, 0

je AandCount

mov ax, A

mov cx, 2

div cx

cmp dx, 1

je AandCount2

jmp after2

AandCount:

inc count

mov ax, A

mov cx, 2

div cx

mov A, ax

jmp after2

AandCount2:

inc count

mov ax, A

mov cx, 3

mul cx

add ax, 1

mov A, ax

jmp after2

break\_count:

inc count

jmp exit\_while2

after2:

jmp while2

exit\_while2:

print "Case "

inc T

mov ax, T

call print\_num

print ": A = "

mov ax, temp

call print\_num

print ", limit = "

mov ax, L

call print\_num

print ", number of terms = "

mov ax, count

call print\_num

printn

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**15. UVA\_694\_C**

#include<stdio.h>

int main()

{

long long int A, L, count, temp;

int T=0;

while(scanf("%lld%lld", &A, &L)==2)

{

if(A<0 && L<0)

break;

temp=A;

count=0;

while(A<=L)

{

if(A==1)

{

count++;

break;

}

else if(A%2==0)

{

count++;

A=A/2;

}

else if(A%2==1)

{

count++;

A=3\*A+1;

}

}

printf("Case %d: A = %lld, limit = %lld,

number of terms = %lld\n", ++T,

temp, L, count);

}

return 0;

}

**16. UVA\_11854\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

a dw ?

b dw ?

c dw ?

x dw ?

y dw ?

z dw ?

ab dw ?

bc dw ?

ca dw ?

flag1 db 0

flag2 db 0

flag3 db 0

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

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

call scan\_num

mov a, cx

printn

call scan\_num

mov b, cx

printn

call scan\_num

mov c, cx

printn

mov ax, a

cmp ax, 0

je check\_b

jmp into\_if

check\_b:

mov ax, b

cmp ax, 0

je check\_c

jmp into\_if

check\_c:

mov ax, c

cmp ax, 0

je else1

into\_if: ;x=a\*a

mov ax, a

mul ax

mov x, ax

;y=b\*b

mov ax, b

mul ax

mov y, ax

;z=c\*c

mov ax, c

mul ax

mov z, ax

;ab=x+y

mov ax, x

add ax, y

mov ab, ax

;ca=x+z

mov ax, x

add ax, z

mov ca, ax

;bc=y+z

mov ax, y

add ax, z

mov bc, ax

;flag1=0, flag2=0, flag3=0

mov flag1, 0

mov flag2, 0

mov flag3, 0

if\_start:

mov ax, x ;check x==bc

cmp ax, bc

je change\_flag1

jmp check\_yAndca

change\_flag1:

mov flag1, 1

check\_yAndca:

mov ax, y ;check y==ca

cmp ax, ca

je change\_flag2

jmp check\_zAndab

change\_flag2:

mov flag2, 1

check\_zAndab:

mov ax, z

cmp ax, ab ;check z==ab

je change\_flag3

jmp if2

change\_flag3:

mov flag3, 1

if2: ;inner if

xor ax, ax

mov al, flag1

or al, flag2

or al, flag3

cmp al, 1

je ifPrint1

else: ;inner else

printn "wrong"

jmp else1

ifPrint1: ;inner if print

printn "right"

else1:

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main prc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

END MAIN

**16.UVA\_11484\_C**

#include<stdio.h>

int main()

{

int a, b, c;

while(scanf("%d %d %d",&a,&b,&c)==3)

{

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

{

int x, y, z, ab, bc, ca;

x=a\*a;

y=b\*b;

z=c\*c;

ab=(a\*a)+(b\*b);

ca=(a\*a)+(c\*c);

bc=(b\*b)+(c\*c);

if(x==bc || y==ca || z==ab)

printf("right\n");

else

printf("wrong\n");

}

}

return 0;

}

**17. UVA\_10079\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while: ;while start

call scan\_num

printn

cmp cx, 0

jl Exit

;piece = (N\*(N+1))/2 + 1

mov ax, cx

add ax, 1

mul cx

mov dl, 2

div dl

add ax, 1

call print\_num

printn

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main prc end

DEFINE\_PRINT\_NUM

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

END MAIN

**17. UVA\_10079\_C**

#include<stdio.h>

int main()

{

long long int N, piece;

while(1)

{

scanf("%lld", &N);

if(N<0)

break;

piece = (N\*(N+1))/2 + 1;

printf("%lld\n", piece);

}

return 0;

}

**18. UVA\_10300\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

s dw ?

i dw ?

j dw ?

k dw ?

a dw ?

b dw ?

c dw ?

l dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ;input i

mov i, cx

printn

mov j, 0

for1:

mov ax, j

cmp ax, i

jge exit\_for1

mov s, 0

call scan\_num ;input k

mov k, cx

mov l, 0

printn

for2:

mov ax, l ;ax=l

cmp ax, k

jge exit\_for2

call scan\_num ;input a

mov a, cx

printn

call scan\_num ;input b

mov b, cx

printn

call scan\_num ;input c

mov c, cx

printn

;s+=a\*c

mov ax, a

mov dx, c

mul dx

add ax, s

mov s, ax

inc l

jmp for2

exit\_for2: ;displaying result

mov ax, s ; ax=s

call print\_num

printn

inc j

jmp for1

exit\_for1:

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM

END MAIN

**18. UVA\_10300\_C++**

#include<stdio.h>

int main()

{

int i,j,a,b,c,k,l,s;

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

{

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

{

s=0;

scanf("%d",&k);

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

{

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

s+=a\*c;

}

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

}

}

return 0;

}

**19. UVA\_10550\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

initial dw ?

first dw ?

second dw ?

third dw ?

g1 dw ?

g2 dw ?

g3 dw ?

.code

main proc

mov ax, @data

mov ds, ax

while: ; while start

CALL SCAN\_NUM ; input initial

mov initial, cx

printn

CALL SCAN\_NUM ; input first

mov first, cx

printn

CALL SCAN\_NUM ; input second

mov second, cx

printn

CALL SCAN\_NUM ; input third

mov third, cx

printn

;if start

mov ax, initial

cmp ax, 0

je check\_first

jmp else

check\_first:

mov ax, first

cmp ax, 0

je check\_second

jmp else

check\_second:

mov ax, second

cmp ax, 0

je check\_third

jmp else

check\_third:

mov ax, third

cmp ax, 0

je Exit:

else:

if1: ; if(initial > first)

mov ax, initial

cmp ax, first

jle else1

mov ax, initial

sub ax, first

mov cx, 9

mul cx

mov g1, ax

jmp if2

else1:

xor dx, dx

mov ax, initial

sub ax, first

add ax, 40

mov cx, 9

mul cx

mov g1, ax

if2: ; if(second>first)

mov ax, second

cmp ax, first

jle else2

mov ax, second

sub ax, first

mov cx, 9

mul cx

mov g2, ax

jmp if3

else2:

mov ax, second

sub ax, first

add ax, 40

mov cx, 9

mul cx

mov g2, ax

if3: ; if(third>second)

mov ax, third

cmp ax, second

jle else3

mov ax, second

sub ax, third

add ax, 40

mov cx, 9

mul cx

mov g3, ax

jmp after

else3:

mov ax, second

sub ax, third

mov cx, 9

mul cx

mov g3, ax

after:

;cout<<720+360+g1+g2+g3<<endl;

mov ax, 1080

add ax, g1

add ax, g2

add ax, g3

CALL PRINT\_NUM\_UNS

printn

jmp while

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM\_UNS

end main

**19. UVA\_10550\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int initial, first, second, third, g1, g2, g3;

while(cin>>initial>>first>>second>>third)

{

if(initial == 0 && first == 0 && second==0 && third ==0 )

break;

else

{

if(initial > first)

g1 = ( initial - first) \* 9; // (360/40)=9

else

g1 = (initial-first+40) \* 9;

if(second>first)

g2=( second-first) \* 9;

else

g2= (second-first+40)\* 9;

if(third>second)

g3 = (second-third+40) \* 9;

else

g3=(second-third)\*9;

cout<<720+360+g1+g2+g3<<endl;

}

}

return 0;

}

**20. UVA\_11000\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

total dw ?

female dw ?

male dw ?

x dw ?

N dw ?

i dw ?

.code

main proc

mov ax, @data

mov ds, ax

while: ;while(scanf("%lld", &N) == 1)

CALL SCAN\_NUM ; input N

mov N, cx

cmp cx, 0

jl after

mov female, 1

mov male, 0

mov total, 1

mov i, 0

printn

for: ; for start

mov ax, i

cmp ax, N

jge print\_res

mov ax, male

mov x, ax ; x = male

mov ax, total

mov male, ax ; male = total

add ax, x

add ax, 1

mov total, ax ; total = male + x + 1

inc i

jmp for

print\_res: ; displaying result

mov ax, male

CALL PRINT\_NUM\_UNS ; display

print " "

mov ax, total

CALL PRINT\_NUM\_UNS ; display

after:

printn

jmp while

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM\_UNS

end main

**20. UVA\_11000\_C**

#include<stdio.h>

int main()

{

long long int female, total, male, x, N, i;

while(scanf("%lld", &N) == 1){

if(N>=0){

female=1; male=0; total=1;

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

{

x = male;

male = total;

total = male + x + 1;

}

printf("%lld %lld\n", male, total);

}

}

return 0;

}

**21. UVA\_11479\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

a dw ?

b dw ?

c dw ?

i dw ?

t dw ?

flag1 db ?

flag2 db ?

flag3 db ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ;scanf("%ld", t);

mov t, cx

mov i, 1

printn

while2: ;while start

mov ax, i

cmp ax, t

jg exit\_while2

call scan\_num ;input a

mov a, cx

printn

call scan\_num

mov b, cx ;input b

printn

call scan\_num ;input c

mov c, cx

printn

mov flag1, 0 ;flag1=0

mov flag2, 0 ;flag1=0

mov flag3, 0 ;flag1=0

mov ax, a

add ax, b

cmp ax, c

jle chg\_fl1

jmp after

chg\_fl1:

mov flag1, 1 ;flag1=1

after:

mov ax, b

add ax, c

cmp ax, a

jle chg\_fl2

jmp after2

chg\_fl2:

mov flag2, 1 ;flag1=1

after2:

mov ax, c

add ax, a

cmp ax, b

jle chg\_fl3

jmp if\_print

chg\_fl3:

mov flag3, 1

xor ax, ax

mov al, flag1 ;al=flag1

or al, flag2 ;flag1 || flag2

or al, flag3 ;flag1 || flag2 || flag33

cmp al, 1

je if\_print

jmp else1

;if print

if\_print:

print "Case "

mov ax, i

call print\_num

print ": Invalid"

printn

jmp last

else1: ;else if1 tart

mov flag1, 0 ;flag1=0

mov flag2, 0 ;flag2=0

mov flag3, 0 ;flag3=0

mov ax, a

cmp ax, 0

jle flag1\_change

jmp sec\_check

flag1\_change:

mov flag1, 1 ;flag1=1

sec\_check:

mov ax, b

cmp ax, 0

jle flag2\_change

jmp third\_check

flag2\_change:

mov flag2, 1 ;flag2=1

third\_check:

mov ax, c

cmp ax, 0

jle flag3\_change

jmp else2

flag3\_change:

mov flag3, 1 ;falg3=1

xor ax, ax

mov al, flag1

or al, flag2

or al, flag3

cmp al, 1

je print\_else1

jmp else2

;else1 print

print\_else1:

print "Case "

mov ax, i

call print\_num

print ": Invalid"

printn

jmp last

else2: ;else2 check

xor ax, ax

mov al, flag1

mov al, flag2

mov al, flag3

mov ax, a

cmp ax, b

je fl1\_change

jmp second\_check

fl1\_change:

mov flag1, 1 ;flag1=0

second\_check:

mov ax, b

cmp ax, c

je fl2\_change

jmp after4

fl2\_change:

mov flag2, 1 ;flag2=0

after4:

xor ax, ax

mov al, flag1

and al, flag2

cmp al, 1

je else2\_print

jmp else3

else2\_print: ;else if2 print

print "Case "

mov ax, i

call print\_num

print ": Equilateral"

printn

jmp last

;else if3

else3:

mov ax, c

cmp ax, a

je fl3\_change

jmp after5

fl3\_change:

mov flag3, 1 ;flag3=1

after5:

xor ax, ax ;ax=0

mov al, flag1

or al, flag2

or al, flag3

cmp al, 1

je else3\_print

jmp else4

else3\_print: ;else if3 print

print "Case "

mov ax, i

call print\_num

print ": Isosceles"

printn

jmp last

else4: ;else print

print "Case "

mov ax, i

call print\_num

print ": Scalene"

printn

last:

inc i

jmp while2 ;jump to while2

exit\_while2:

jmp while ;jump to while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**21. UVA\_11479\_C++**

#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;

}

**22. UVA\_11727\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ;data section

t dw ?

i dw ?

first dw ?

sec dw ?

third dw ?

Survives dw ?

large dw ?

min dw ?

.code ;code section

main proc ;main procedure

mov ax, @data ;import data

mov ds, ax

while: ;while start

call scan\_num ;input t

mov t, cx

printn

mov i, 1 ;i=1

for: ;for start

mov ax, i

cmp ax, t

jg exit\_for

;input firstEmployee

call scan\_num

mov first, cx

printn

call scan\_num ;input secEmployee

mov sec, cx

printn

call scan\_num ;input thirdEmployee

mov third, cx

printn

m1: mov ax, first ;find large

cmp ax, sec ; check first and sec

jg check1And3Max

mov ax, sec

cmp ax, third ;check sec and third

jg large\_sec

mov ax, third

mov large, ax ;large=third

jmp find\_min

check1And3Max: ;check first and third

mov ax, first

cmp ax, third

jg large\_first

jmp find\_min

large\_first:

mov ax, first

mov large, ax ;large=first

jmp find\_min

large\_sec:

mov ax, sec

mov large, ax ;large=sec

find\_min: ;find min

mov ax, first

cmp ax, sec ;check first and sec

jl check1And3Min

mov ax, sec

cmp ax, third ;check sec and third

jl min\_sec

mov ax, third

mov min, ax ;min=third

jmp survive

check1And3Min:

mov ax, first

cmp ax, third ;check first and third

jl first\_min

mov ax, third

mov min, ax ;min=third

jmp survive

first\_min:

mov ax, first

mov min, ax ;min=first

jmp survive

min\_sec:

mov ax, sec

mov min, ax ;min=sec

survive:

mov ax, first ;ax=first

add ax, sec ;ax+=sec

add ax, third ;ax+=third

sub ax, large ;ax-=large

sub ax, min ;ax-=min

mov survives, ax ;result

;displaying result

print "Case "

mov ax, i

call print\_num\_uns

print ": "

mov ax, survives

call print\_num\_uns

printn

inc i

jmp for

exit\_for:

jmp while

mov ah, 4ch

int 21h

main endp

define\_scan\_num

define\_print\_num\_uns

end main

**22. UVA\_11727\_C++**

#include<stdio.h>

int main()

{

int T, i, firstEmployee, secEmployee, thirdEmployee, Survives, m1,large, small, m2;

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

{

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

{

scanf("%d%d%d", &firstEmployee, &secEmployee, &thirdEmployee);

m1 = (firstEmployee>secEmployee)? firstEmployee:secEmployee;

large = (m1>thirdEmployee)? m1:thirdEmployee;

m1 = (firstEmployee<secEmployee)? firstEmployee:secEmployee;

small = (m1<thirdEmployee)? m1:thirdEmployee;

Survives = (firstEmployee+secEmployee+thirdEmployee)-(large+small);

printf("Case %d: %d\n", i, Survives);

}

}

return 0;

}

**23. UVA\_11877\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

temp dw ?

flag dw ?

e dw ?

N dw ?

ans dw ?

.code

main proc ; main proceure

mov ax, @data ; data import

mov ds, ax

while1: ; while(scanf("%d", &N)==1)

CALL SCAN\_NUM ; input N

MOV N, CX

add cx, 1

mov temp, cx

printn ; print newline

mov flag, 1 ; flag=1

mov e, 0 ; e=0

xor ax, ax ; ax=0

mov ax, N ; ax=N

cmp ax, 0

jne else

mov flag, 1

jmp after

else: ; else

while2: ; while(temp>=3)

;e=e+(temp/3)

xor dx, dx

mov ax, temp

mov cx, 3

div cx

mov ans, ax ; ans=temp/3

add ax, e

mov e, ax

;temp=(temp%3)+(temp/3)

xor ax, ax

mov ax, ans

add ax, dx

mov temp, ax

mov flag, 0 ; flag=0

xor ax, ax ; ax=0

mov ax, temp

cmp ax, 3

jge while2 ; jump to while2

xor ax, ax ; if(flag==0)

mov ax, flag

cmp ax, 0

je print\_e

jmp after

print\_e: ; printf("%d\n", e);

mov ax, e

CALL PRINT\_NUM\_UNS ; displaying result

after: ; print newline

printn

jmp while1 ; jump to while1

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_SCAN\_NUM ; define build in function

DEFINE\_PRINT\_NUM\_UNS

end main

**23. UVA\_11877\_C++**

#include<stdio.h>

int main(){

int N;

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

{

int temp=N+1;

int flag=1;

int e=0;

if(N==0){

flag=1;

}

else{

while(temp>=3)

{

e=e+(temp/3);

temp=(temp%3)+(temp/3);

flag=0;

}

if(flag==0){

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

}

}

}

return 0;

}

**24. UVA\_11936\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

N dw ?

i dw ?

a dw ?

b dw ?

c dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ;input N

mov N, ax ;N=ax

mov i, 1 ;i=1

printn

for: ;for start

mov ax, i

cmp ax, N

jg exit\_for

call scan\_num ;input a

mov a, cx

printn

call scan\_num ;input b

mov b, cx

printn

call scan\_num ;input c

mov c, cx

printn

;if((a+b)>c)

mov ax, a

add ax, b

cmp ax, c

jle else

printn "OK"

jmp after

else: ; else

printn "Wrong!!"

after:

inc i

jmp for

exit\_for:

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_SCAN\_NUM

END MAIN

**24. UVA\_11936\_C**

#include<stdio.h>

int main()

{

int a, b, c, i, N;

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

{

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

{

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

if((a+b)>c)

printf("OK\n");

else

printf("Wrong!!\n");

}

}

return 0;

}

**25. UVA\_12157\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

T dw ?

N dw ?

sec dw ?

amMile dw ?

amJuice dw ?

i dw ?

j dw ?

temp dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

call scan\_num ;cin>>T

mov T, cx

printn

mov i, 1

for: ;for(int i=1; i<=T; i++)

mov ax, i

cmp ax, T

jg Exit

call scan\_num ;cin>>N

mov N, cx

printn

mov amMile, 0 ;amMile=0

mov amJuice, 0 ;amJuice=0

mov j, 1

for2:

mov ax, j

cmp ax, N

jg exit\_for2

call scan\_num ;cin>>second

mov sec, cx

printn

mov ax, sec ;temp=second

mov temp, ax

xor dx, dx ;amMile+=(sec/30)\*10+10

mov ax, sec

mov cx, 30

div cx

mov cx, 10

mul cx

add ax, 10

add ax, amMile

mov amMile, ax

xor dx, dx ;amJuice+=(sec/60)\*15+15

mov ax, sec

mov cx, 60

div cx

mov cx, 15

mul cx

add ax, 15

add ax, amJuice

mov amJuice, ax

inc j

jmp for2

exit\_for2: ;if(amountJuice==amountMile)

mov ax, amMile

cmp ax, amJuice

je first\_print

mov ax, amJuice

cmp ax, amMile

jg sec\_print ;else if(amountJuice>amountMile)

print "Case " ;cout<<"Case "<<i<<": Juice "<<amountJuice<<endl;

mov ax, i

call print\_num

print ": Juice "

mov ax, amJuice

call print\_num

printn

jmp after

;cout<<"Case "<<i<<": Mile Juice "<<amountJuice<<endl

first\_print:

print "Case "

mov ax, i

call print\_num

print ": Mile Juice "

mov ax, amJuice

call print\_num

printn

jmp after:

;cout<<"Case "<<i<<": Mile

;"<<amountMile<<endl;

sec\_print:

print "Case "

mov ax, i

call print\_num

print ": Mile "

mov ax, amMile

call print\_num

printn

jmp after

after:

inc i

jmp for

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**25. UVA\_12157\_C++**

#include<bits/stdc++.h>

#define CSEKU\_160212 main()

using namespace std;

int CSEKU\_160212

{

int T, N, second;

int amountMile, amountJuice;

cin>>T;

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

{

cin>>N;

amountMile=0;

amountJuice=0;

for(int j=1; j<=N; j++)

{

cin>>second;

int temp=second;

amountMile+=(second/30)\*10+10;

amountJuice+=(second/60)\*15+15;

}

if(amountJuice==amountMile)

cout<<"Case "<<i<<": Mile Juice "<<amountJuice<<endl;

else if(amountJuice>amountMile)

cout<<"Case "<<i<<": Mile "<<amountMile<<endl;

else

cout<<"Case "<<i<<": Juice "<<amountJuice<<endl;

}

return 0;

}

**26. UVA\_12468\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

a dw ?

b dw ?

clickForward dw ?

clickBackward dw ?

max dw ?

min dw ?

.CODE ;code section

MAIN PROC ;main proc start

MOV AX, @DATA ;import data

MOV DS, AX

while:

call scan\_num ;cin>>a

mov a, cx

printn

call scan\_num ;cin>b

mov b, cx

printn

mov ax, a ;find max

cmp ax, b

jge max\_ini

mov ax, b

mov max, ax ;max=b

jmp after

max\_ini:

mov ax, a ;max=a

mov max, ax

after:

mov ax, a ;find min

add ax, b ;ax=a+b

sub ax, max ;ax=(a+b)-max

mov min, ax ;min=ax

mov ax, max ;clickForward=max-min

sub ax, min

mov clickForward, ax

mov ax, 100 ;clickBackward=100-clickForward;

sub ax, clickForward

mov clickBackward, ax

mov ax, clickForward

cmp ax, clickBackward

jge if\_print

mov ax, clickForward ;cout<<clickForward;

call print\_num

printn

jmp after2

if\_print:

mov ax, clickBackward

call print\_num ;cout<<clickBackward;

printn

after2:

jmp while ;jump to while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**26. UVA\_12468\_Assembly**

#include<bits/stdc++.h>

#define CSEKU\_160212 main()

using namespace std;

int CSEKU\_160212

{

int a, b, clickForward, clickBackward;

while(cin>>a>>b)

{

if(a==-1 && b==-1)

break;

clickForward=max(a,b)-min(a,b);

clickBackward=100-(max(a,b)-min(a,b));

if(clickForward>=clickBackward)

cout<<clickBackward<<endl;

else

cout<<clickForward<<endl;

}

Return 0 ;

}

**27. UVA\_12696\_Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data ; data section

length dw ?

width dw ?

depth dw ?

weight dw ?

count dw ?

i dw ?

t dw ?

flag1 dw ?

flag2 dw ?

.code ; code section

main proc

mov ax, @data ; import data

mov ds, ax

while: ;while(scanf("%d", &t)==1)

CALL SCAN\_NUM ; input t

mov t, cx

printn

mov count, 0

mov i, 0

for:

CALL SCAN\_NUM ; input lendth

MOV length, CX

printn

CALL SCAN\_NUM ; input width

MOV width, CX

printn

CALL SCAN\_NUM ; input depth

MOV depth, CX

printn

CALL SCAN\_NUM ; input weight

MOV weight, CX

printn

mov flag1, 0 ; flag1=0

mov flag2, 0 ; flag2=0

mov ax, length ; check length

cmp ax, 56

jle width\_check

jmp after

width\_check: ; check width

mov ax, width

cmp ax, 45

jle check\_depth

jmp after

check\_depth: ; check depth

mov ax, depth

cmp ax, 25

jle flag1\_change

jmp after

flag1\_change: ; flag1 change

mov flag1, 1

after: ; second check for flag2

mov ax, length ; ax=length

add ax, width ; ax+=width

add ax, depth ; ax+=depth

cmp ax, 125 ; length+width+depth)<=125

jle check\_weight

jmp after2

check\_weight: ; weigth check

mov ax, weight

cmp ax, 7

jle flag2\_change

jmp after2

flag2\_change: ; flag2 change

mov flag2, 1

after2:

mov ax, flag1 ; ax=flag1

or ax, flag2 ; flag1 || flag2

cmp ax, 0

je else

printn "1" ; printf("1\n")

inc count ; count++

jmp after3

else: ; printf("0\n")

printn "0"

after3:

inc i ; i++

mov ax, i

cmp ax, t ; check i and t

jge print\_count

jmp for

print\_count: ; printf("%d\n", count)

mov ax, count

CALL PRINT\_NUM\_UNS

printn

jmp while ; jumpt to while

Exit:

mov ah, 4ch

int 21h

main endp

DEFINE\_SCAN\_NUM

DEFINE\_PRINT\_NUM\_UNS

end main

**27. UVA\_12696\_C++**

#include<stdio.h>

int main()

{

int t, i, count;

double length, width, depth, weight;

while(scanf("%d", &t)==1){

count=0;

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

{

scanf("%lf%lf%lf%lf", &length, &width, &depth, &weight);

if(((length<=56 && width<=45 && depth<=25) || (length+width+depth)<=125) && weight<=7.00)

{

printf("1\n");

count++;

}

else

printf("0\n");

}

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

}

return 0;

}

**28. UVA\_12917\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

x dw ?

y dw ?

z dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ; input x

mov x, cx

printn

call scan\_num ;input y

mov y, cx

printn

call scan\_num ;input z

mov z, cx

printn

mov ax, z ; ax=z

sub ax, y ; ax-=y

cmp ax, x

jl else

printn "Props win!" ; if

jmp after

else:

printn "Hunters win!" ;else

after:

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**28. UVA\_12917\_C++**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int x,y,z;

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

{

if(x<=(z-y))

cout<<"Props win!"<<endl;

else

cout<<"Hunters win!"<<endl;

}

return 0;

}

**29. UVA\_13012\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

c dw ?

i dw ?

n dw ?

t dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

while:

call scan\_num ; input t

mov t, cx

printn

mov i, 1 ;i=1

mov n, 0 ;n=0

for:

mov ax, i

cmp ax, 5

jg exit\_for

call scan\_num

mov c, cx

printn

mov cx, c ;compare c and t

cmp cx, t

je nInc

jmp for\_jump

nInc:

inc n

for\_jump:

inc i

jmp for

exit\_for:

mov ax, n

call print\_num

printn

jmp while

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**29. UVA\_13012\_C++**

#include <bits/stdc++.h>

using namespace std;

int main()

{

int t,c,i,n;

while(cin>>t)

{

n=0;

for(i=1;i<=5;i++)

{

cin>>c;

if(c==t)

{

n++;

}

}

cout<<n<<endl;

}

return 0;

}

**30. UVA\_11388\_Assembly**

INCLUDE 'emu8086.inc' ; include library function

.MODEL SMALL

.STACK 100H

.DATA ; data section

i dw ?

j dw ?

g dw ?

l dw ?

.CODE ; code section

MAIN PROC ; main proc start

MOV AX, @DATA ; import data

MOV DS, AX

call scan\_num

mov i, cx ;i=cx

printn

mov j, 1 ;j=1

for:

mov ax, j ;ax=j

cmp ax, i

jg Exit

call scan\_num

mov g, cx

printn

call scan\_num

mov l, cx

if:

xor dx, dx

mov ax, l

mov cx, g

mul cx

cmp dx, 0

je print\_if

else:

printn "-1"

jmp after

print\_if:

mov ax, g

call print\_num

print " "

mov ax, l

call print\_num

printn

after:

inc j

jmp for

Exit:

MOV AH, 4CH ; exit

INT 21H

MAIN ENDP ; main proc end

DEFINE\_PRINT\_NUM\_UNS

DEFINE\_PRINT\_NUM

DEFINE\_SCAN\_NUM

END MAIN

**30. UVA\_11388\_C++**

#include<stdio.h>

int main()

{

int i,j,g,l;

scanf("%d",&i);

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

{

scanf("%d %d",&g,&l);

if(l%g==0)

{

printf("%d %d\n",g,l);

}

else

{

printf("-1\n");

}

}

return 0;

}