**UVA 136 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.code

main proc

printn "The 1500'th ugly number is 859963392."

mov ah,4ch

int 21h

end main

end main

**UVA 136 C**

#include<stdio.h>

int main()

{

printf("The 1500'th ugly number is 859963392.\n");

return 0;

}

**UVA 264 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

info db "The input list contains a single number per line and",10,13," will be terminated by 0.$"

value db ?

a db ?

b db ?

i db ?

n db ?

.code

main proc

;fetching data segment

mov ax,@data

mov ds,ax

;printing info

lea dx,info

mov ah,9

int 21h

printn

mov value,0 ;value=0

again:

input\_for:

;user input

mov ah,1

int 21h

cmp al,13 ;checking input==\n

je next\_loop

;taking multi digit input

sub al,48

mov bl,al

mov al,value

mov ah,10

mul ah

add al,bl

mov value,al

jmp input\_for

next\_loop:

mov al,value ;checking input==0?

cmp al,0

je exit ;if true then exit

mov value,al

mov n,al

mov bh,1 ;for(int i=1

;making the operation with for loop

for\_loop:

mov bl,value

cmp bh,bl ;i<value

jge next

sub bl,bh ;value-i

mov value,bl

inc bh ;i++

mov i,bh

jmp for\_loop

next:

;getting the mod value of i%2 store in ah

mov al,i

mov bl,2

div bl

cmp ah,1 ;checking ah==1?

je work1

mov al,value

mov a,al ;a=value

jmp work2

work1:

;a=1+i-value

mov bl,1

mov bh,i

add bl,bh

sub bl,value

mov a,bl

work2:

; b=i-a+1;

mov bh,i

mov bl,a

sub bh,bl

add bh,1

mov b,bh

;printing the result in this formation

;printf("TERM %d IS %d/%d\n",n,a,b);

printn

print "TERM "

mov bh,n

cmp bh,9

jg twodigit\_value

mov ah,2

mov dl,n

add dl,48

int 21h

jmp is

twodigit\_value:

mov al,0

add al,n

mov ah,0

aaa

mov bx,ax

mov ah,2

mov dl,bh

add dl,48

int 21h

mov dl,bl

add dl,48

int 21h

is:

print " IS "

mov ah,2

mov dl,a

add dl,48

int 21h

print "/"

mov ah,2

mov dl,b

add dl,48

int 21h

printn

;making all variable value to 0

mov value,0

mov i,0

mov a,0

mov b,0

jmp again ;calling again for restarting the program

exit:

mov ah,4ch

int 21h

main endp

end main

**UVA 264 C**

#include<stdio.h>

int main()

{

int a,b,n,i,value;

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

{

value=n;

for(i=1; value>i; i++)

value-=i;

if(i%2==1)

a=1+i-value;

else

a=value;

b=i-a+1;

printf("TERM %d IS %d/%d\n",n,a,b);

}

return 0;

}

**UVA 382 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

value db ?

result db ?

length db ?

.code

main proc

;fetching data from data segment

mov ax,@data

mov ds,ax

printn "PERFECTION OUTPUT"

again:

;making all the variable value to zero

mov result,0

mov value,0

;taking input

;scanf for two digit

loop\_input:

mov ah,1

int 21h

cmp al,13 ;checking whether it is new line

je next

sub al,48

mov bl,al

mov al,value

mov bh,10

mul bh

add al,bl

mov value,al

jmp loop\_input

next:

printn

;input value/2

mov al,value

cmp al,0

je exit

mov ah,0

mov bl,2

div bl

mov length,al

mov bl,1 ;for(int i=1

loop\_for:

cmp bl,length ;i<value/2

jg real\_ans

mov al,value

mov ah,0

div bl

cmp ah,0

je addition ;jumping addition loop if(value%2==0)

inc bl ;i++

jmp loop\_for

;result+=i;

addition:

mov bh,result

add bh,bl

mov result,bh

inc bl

jmp loop\_for

real\_ans:

mov bh,result

cmp bh,value

je equal ;if(sum==a) jmp equal

cmp bh,value ;if(sum<a)

jl lesser ;jmp lesser

;else printf("%5d ABUNDANT\n",a);

mov bh,value

cmp bh,9

jg greater1

mov ah,2

mov dl,value

add dl,48

int 21h

printn " ABUNDANT"

jmp again

;for two digit output

greater1:

mov al,value

mov ah,0

aaa

mov bx,ax

mov ah,2

mov dl,bh

add dl,48

int 21h

mov dl,bl

add dl,48

int 21h

printn " ABUNDANT"

jmp again

;printf("%5d PERFECT\n",a);

equal:

mov bh,value

cmp bh,9

jg greater2

mov ah,2

mov dl,value

add dl,48

int 21h

printn " PERFECT"

jmp again

;for two digit output

greater2:

mov al,value

mov ah,0

aaa

mov bx,ax

mov ah,2

mov dl,bh

add dl,48

int 21h

mov dl,bl

add dl,48

int 21h

printn " PERFECT"

jmp again

jmp again

printf("%5d DEFICIENT\n",a);

lesser:

mov bh,value

cmp bh,9

jg greater3

mov ah,2

mov dl,value

add dl,48

int 21h

printn " DEFICIENT"

jmp again

;for two digit output

greater3:

mov al,value

mov ah,0

aaa

mov bx,ax

mov ah,2

mov dl,bh

add dl,48

int 21h

mov dl,bl

add dl,48

int 21h

printn " DEFICIENT"

jmp again

;jump exit

exit:

mov ah,4ch

int 21h

main endp

end main

**UVA 382 C**

#include<stdio.h>

int main()

{

int a,sum,i;

printf("PERFECTION OUTPUT\n");

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

{

sum=0;

if(a==0){

printf("END OF OUTPUT\n");

return 0;

}

for(i=1;i<=a/2;i++)

{

if(a%i==0)

sum=sum+i;

}

if(sum==a)

printf("%5d PERFECT\n",a);

else if(sum<a)

printf("%5d DEFICIENT\n",a);

else

printf("%5d ABUNDANT\n",a);

}

return 0;

}

**UVA 488 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

case db ?

limit db ?

line db ?

i db ?

j db ?

k db ?

l db ?

m db ?

n db ?

.code

main proc

;fetching data fro data segment

mov ax,@data

mov ds,ax

again:

mov i,1 ;for(i=1;

mov j,1 ;for(j=1;

mov k,1 ;for(k=1;

mov l,1 ;for(l=1;

mov n,1 ;for(n=1;

mov dl,0

call input

mov case,dl ;scanf("%d",&t)

printn

case\_for:

mov bl,i

cmp bl,case ;i<=t;

jg exit

mov dl,0

call input

mov limit,dl ;scanf("%d",&a);

printn

mov bl,limit

sub bl,1

mov m,bl ;for(m=limit-1;

mov dl,0

call input

mov line,dl ;scanf("%d",&b);

printn

main\_for:

mov bl,j

cmp bl,line ;j<=b;

jg case\_for\_inc ;if j>b then case\_for\_inc

in\_for\_1:

mov bl,k

cmp bl,limit ;k<=a;

jg in\_for\_2 ;if k>a then in\_for\_2

in\_in\_for\_1:

mov bl,l

cmp bl,k ;l<=k;

jg in\_for\_1\_inc ;if l>k then in\_for\_1\_inc

mov dl,k

add dl,48 ;printf("%d",k);

mov ah,2

int 21h

inc l ;l++)

jmp in\_in\_for\_1

in\_for\_1\_inc:

mov l,1

printn

inc k ;k++)

jmp in\_for\_1

in\_for\_2:

mov bl,m

cmp bl,1 ;m>=1;

jl main\_for\_inc

in\_in\_for\_2:

mov bl,n

cmp bl,m ; n<=m;

jg in\_for\_2\_dec ;if n>m then in\_for\_2\_dec

mov dl,m

add dl,48 ;printf("%d",m);

mov ah,2

int 21h

inc n ;n++)

jmp in\_in\_for\_2

in\_for\_2\_dec:

mov n,1

printn

dec m ;m--)

jmp in\_for\_2

main\_for\_inc:

mov bl,i

cmp bl,case ;if(k!=t ||

jne new\_line ;jmp new\_line

mov bl,j

cmp bl,line ;|| l!=b)

jne new\_line ;jmp new\_line

jmp real\_main\_for\_inc ;else

;jmp real\_main\_for\_in new\_line:

printn ;\n

real\_main\_for\_inc:

mov k,1

mov bl,limit

sub bl,1

mov m,bl

inc j ;j++)

jmp main\_for

case\_for\_inc:

mov j,1

inc i ;i++)

jmp case\_for

exit:

mov ah,4ch

int 21h ;return 0;

main endp

input proc

loop\_input:

mov ah,1

int 21h ;checking for new line

cmp al,13

je loop\_exit

sub al,48

mov bl,al

mov al,dl

mov ah,0 ;dl=1 or 2 digit input

mov bh,10

mul bh

add al,bl

mov dl,al

jmp loop\_input

loop\_exit:

ret

input endp

end main

**UVA 488 C**

#include<stdio.h>

int main()

{

int t,a,b,i,j,k,l;

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

{

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

{

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

for(l=1; l<=b; l++)

{

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

{

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

{

printf("%d",j);

}

printf("\n");

}

for(j=a-1; j>=1; j--)

{

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

{

printf("%d",j);

}

printf("\n");

}

if(k!=t || l!=b)

printf("\n");

}

}

}

return 0;

}

**UVA 568 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

sum db ?

i db ?

input db ?

.code

main proc

;fetching data fro data segment

mov ax,@data

mov ds,ax

printn "this only works for 1 to 11 input because"

printn "after this kind of input makes overflow to the register"

again:

mov input,0

;taking the input to the 2 digit

loop\_input:

mov ah,1

int 21h

cmp al,13

je loop\_exit1

sub al,48

mov bl,al

mov al,input

mov ah,0

mov bh,10

mul bh

add al,bl

mov input,al

jmp loop\_input

;after takinng input checking whether the input value is null

loop\_exit1:

mov al,input

cmp al,0

je loop\_input

mov sum,1

mov i,1

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

loop\_for:

mov al,input

cmp al,i

jl loop\_exit

mov al,sum

mov bl,i ;sum=sum\*i;

mul bl

mov sum,al

inc i

;while(sum%10==0)

loop\_while:

mov al,sum

mov ah,0

mov bl,10 ;sum/=10;

div bl

cmp ah,0

jg loop\_for ;if ==0 then again go to for loop

mov sum,al

jmp loop\_while

jmp loop\_for

loop\_exit:

printn

mov al,sum

mov ah,0

mov bl,10 ;sum=sum%10;

div bl

mov sum,ah

;printf("%5d -> %lld\n",a,sum);

mov bl,input

cmp bl,9

jg greater

mov dl,input

add dl,48

mov ah,2 ;for 1 digit output

int 21h

jmp result

greater:

mov al,input

mov ah,0

aaa

mov bx,ax

mov dl,bh

add dl,48 ;for 2 digit output

mov ah,2

int 21h

mov dl,bl

add dl,48

mov ah,2

int 21h

result:

print " -> "

mov bl,sum

cmp bl,9

jg greater2

mov dl,sum

add dl,48

mov ah,2 ;for 1 digit output

int 21h

jmp restart

greater2:

mov al,sum

mov ah,0

aaa

mov bx,ax

mov dl,bh

add dl,48 ;for 2 digit output

mov ah,2

int 21h

mov dl,bl

add dl,48

mov ah,2

int 21h

restart:

printn

jmp again

exit:

mov ah,4ch

int 21h ;return 0;

main endp

end main

**UVA 568 C**

#include<stdio.h>

int main()

{

int a,i;

long long sum;

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

{

sum=1;

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

{

sum=sum\*i;

while(sum%10==0)

sum/=10;

sum%=100000;

}

sum=sum%10;

printf("%5d -> %lld\n",a,sum);

}

return 0;

}

**UVA 591 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

t dw ?

sum dw ?

sum2 dw ?

i dw ?

n dw ?

m dw ?

j dw ?

value dw ?

count dw ?

a dw 100 dup(0)

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov j,1

mov count,0 ;count=0

mov value,0

call input

mov ax,value ;scanf("%d",&t)!=

mov t,ax

printn

cmp ax,0

je exit ;checking whetther t!=0

mov sum,0 ;sum=0;

mov sum2,0 ;sum2=0;

mov i,0 ;for(i=0;

mov si,0 ;array index

for1:

mov ax,i

cmp ax,t

jge for2\_zero ;i<T;

mov value,0

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

mov ax,value

printn

mov a[si],ax

add ax,sum

mov sum,ax ;sum+=a[i];

inc si ; as define word then index+=2

inc si

inc i ;i++)

jmp for1

for2\_zero:

xor dx,dx

mov ax,sum

mov bx,t

div bx

mov n,ax ;n=sum/t;

mov i,0 ;for(i=0;

mov si,0 ;array index

for2:

mov ax,i

cmp ax,t

jge push\_before ;i<T;

mov ax,n

cmp a[si],ax

jle increament ;if(a[i]>n)

mov ax,a[si]

sub ax,n

mov m,ax ; m=a[i]-n;

mov ax,m

add sum2,ax ;sum2+=m;

increament:

inc si ; as define word then index+=2

inc si

inc i ;i++)

jmp for2

push\_before:

print "Set#" ;printf("Set #

push\_value:

mov ax,j

cmp ax,0 ;checking for j is 0 or not

je pop\_value

xor dx,dx

mov bx,10

div bx ;sum/10

push dx ;pushing last digit as reminder

mov j,ax

inc count ;value length increase

jmp push\_value

pop\_value:

mov ax,count

cmp ax,0 ;checking for value length

je push\_before2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value

push\_before2:

printn ;printf("The minimum number of moves is")

print "The minimum number of moves is "

push\_value2:

mov ax,sum2

cmp ax,0 ;checking for sum2 is 0 or not

je pop\_value2

xor dx,dx

mov bx,10

div bx ;sum/10

push dx ;pushing last digit as reminder

mov sum2,ax

inc count ;value length increase

jmp push\_value2

pop\_value2:

mov ax,count

cmp ax,0 ;checking for value length

je exit2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value2

exit2:

printn

inc j ;j++;

jmp again ;calliing the program again

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx

push cx ;saving all data if used

push dx

for\_loop:

mov ah,1

int 21h ;getting input

cbw

cmp ax,13

je exit3 ;checking whether it is new line

cmp ax,32

je exit3 ;checking whether it is space

sub ax,48 ;making pure digit

mov cx,ax ;cx=input

mov ax,value

mov bx,10 ;value=value\*10

mul bx

add ax,cx ;recent result+input

mov value,ax ;value=recent result

jmp for\_loop ;loop call

exit3:

pop dx

pop cx

pop bx ;restoring all registor value

pop ax

ret

input endp

end main

**UVA 591 C**

#include<stdio.h>

int main()

{

int T,sum,sum2,i,n,m,j=1;

while(scanf("%d",&T)==1 && T!=0)

{

sum=0;

sum2=0;

int a[T];

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

{

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

sum+=a[i];

}

n=sum/T;

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

{

if(a[i]>n)

{

m=a[i]-n;

sum2+=m;

}

}

printf("Set #%d\nThe minimum number of moves is %d.\n\n",j,sum2);

j++;

}

return 0;

}

**UVA 900 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

testcase db ?

a db ?

b db ?

.code

main proc

;fetching data fro data segment

mov ax,@data

mov ds,ax

again:

;making the value set

mov a,0

mov b,1

;taking the testcase input while (testcase!=0)

mov ah,1

int 21h

sub al,48

cmp al,0

je exit

mov testcase,al

mov al,0 ;for(int i=0

for\_start:

cmp al,testcase ;i<testcase

jge result

inc al ;i++

mov bh,b ;c=b;

mov bl,a

add bl,bh ;b=a+b;

mov b,bl ;assign to b

mov a,bh ;a=c;

jmp for\_start

result:

printn

mov bl,b

cmp bl,9 ;printf("%d\n",b); for 1 digit

jg greater

mov dl,b

add dl,48

mov ah,2

int 21h

printn

jmp again

greater:

mov al,b

mov ah,0

aaa

mov bx,ax

mov dl,bh ;printf("%d\n",b); for 2 digit

add dl,48

mov ah,2

int 21h

mov dl,bl

add dl,48

int 21h

printn

jmp again

exit:

mov ah,4ch

int 21h

main endp

end main

**UVA 900 C**

#include<stdio.h>

int main()

{

int t,i;

while(scanf("%d",&t)==1 && t!=0)

{

int a=0,b=1,sum=0,c;

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

{

c=b;

b=a+b;

a=c;

}

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

}

return 0;

}

**UVA 913 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

input\_value db ?

i db ?

j db ?

k db ?

sum db ?

final db ?

.code

main proc

mov ax,@data

mov ds,ax

again:

mov dl,0

call input

mov input\_value,dl ;scanf("%lld",&n)

printn

mov j,1 ;j=1

mov sum,0 ;sum=0

mov i,1 ;for(i=1;

for1:

mov bl,i

cmp bl,input\_value ;i<input\_value;

jge final\_get

inc j ;j++

inc i ;i+=2)

inc i

jmp for1

final\_get:

;final=(j\*i)+j-1;

mov al,j ;al=j

mov ah,0 ;ah=0

mov bl,i ;bl=i

mul bl ;al=al\*bl

add al,j ;al+=j

sub al,1 ;al-=1

mov final,al ;final=al

mov k,1 ;for(k=1;

for2:

mov bl,k

cmp bl,3 ;k<=3;

jg print\_sum

mov al,sum

add al,final

mov sum,al ;sum=sum+final;

dec final ;final-=2;

dec final

inc k ;k++)

jmp for2

print\_sum:

mov dl,sum

cmp dl,9

jg greater

add dl,48

mov ah,2 ;printf("%lld\n",sum); for 1 digit output

int 21h

jmp jump\_again

greater:

mov al,sum

mov ah,0

mov bl,10

div bl

mov bx,ax

mov dl,bl

mov ah,2

add dl,48 ;printf("%lld\n",sum); for 2 digit output

int 21h

mov dl,bh

add dl,48

int 21h

jump\_again:

printn

jmp again

exit:

mov ah,4ch

int 21h

main endp

input proc

loop\_input:

mov ah,1

int 21h ;checking for new line

cmp al,13

je loop\_exit

sub al,48

mov bl,al

mov al,dl

mov ah,0 ;dl=1 or 2 digit input

mov bh,10

mul bh

add al,bl

mov dl,al

jmp loop\_input

loop\_exit:

ret

input endp

end main

**UVA 913 C**

#include<stdio.h>

int main()

{

long long int n;

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

{

long long int i,j=1,finall,sum=0,k;

for(i=1;i<n;i+=2)

j++;

finall=(j\*i)+j-1;

for(k=1;k<=3;k++)

{

sum=sum+finall;

finall-=2;

}

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

}

return 0;

}

**UVA 1124 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

str1 db 255

.code

main proc

lea si,str1

input:

mov ah,1

int 21h

cmp al,13 ;gets(a)

je print

mov [si],al

inc si

jmp input

print:

printn

inc si

mov dl,'$' ;setting a last finish indicator in the last of the string

mov [si],dl

lea dx,str1

mov ah,9 ;printf("%s\n",a);

int 21h

printn

jmp input

main endp

end main

**UVA 1124 C**

#include<stdio.h>

#include<string.h>

int main()

{

char a[100000];

while(gets(a))

{

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

}

return 0;

}

**UVA 10035 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

value1 dw ?

value2 dw ?

i dw ?

carry dw ?

r dw ?

c dw ?

value dw ?

.code

main proc

;fetching the data

mov ax,@data

mov ds,ax

restart:

;calling for first multidigit number

mov value,0

call input ;input function call

mov dx,value

mov value1,dx ;value1

printn ;new line

;calling for 2nd multidigit number

mov value,0

call input

mov dx,value

mov value2,dx

mov ax,value1

cmp ax,0 ;if (value1==0

je second\_zero ;then go for second value checking

jmp main\_work

second\_zero:

mov ax,value2

cmp ax,0 ;&& value2==0)

je exit ;then jump exit

main\_work:

mov r,0 ;r=0;

mov carry,0 ;carry=0;

mov i,1 ;for(i=1;

main\_for:

mov ax,value1

cmp ax,0

je second\_zero2

jmp work

second\_zero2:

mov ax,value2 ;if (value1==0 && value2==0)

cmp ax,0 ;then go for printing result

je print\_result

work:

mov c,0

mov ax,i ;for(i=1; i<16;

cmp ax,16

jge print\_result

mov ax,r

add c,ax ;c+=r;

xor dx,dx

mov ax,value1

mov bx,10

div bx

add c,dx ;c+= (value1%10)

xor dx,dx

mov ax,value2

mov bx,10

div bx

add c,dx ;c+=(value2%10)

mov ax,c

cmp ax,9 ;if(c>9)

jg r\_1 ;then go to r\_1

mov r,0 ;else r=0

jmp division\_values

r\_1:

inc carry ;carry++;

mov r,1 ;r=1;

division\_values:

xor dx,dx

mov ax,value1

mov bx,10

div bx

mov value1,ax ;value1/=10;

xor dx,dx

mov ax,value2

mov bx,10

div bx

mov value2,ax ;value2/=10;

inc i ;i++)

jmp main\_for

print\_result:

printn

mov ax,carry

cmp ax,0 ;if(carry==0)

je no\_carry

mov ax,carry

cmp ax,1 ;else if(carry==1)

je one\_carry

mov dx,carry ;else

add dl,48 ;print carry value

mov ah,2

int 21h

printn " carry operation." ;printf("%d carry operations.\n",carry);

jmp restart

no\_carry:

printn "No carry operation." ;printf("No carry operation.\n");

jmp restart

one\_carry:

printn "1 carry operation." ;printf("1 carry operation.\n");

jmp restart

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx ;taking all register in stack

push cx

push dx

input\_for:

mov ah,1

int 21h ;scanf

cmp al,13

je exit\_for

sub al,48

cbw ;converting byte to word

mov cx,ax

mov ax,value

mov bx,10 ;saving value

mul bx

add ax,cx

mov value,ax ;value=value\*10 + scanf

jmp input\_for

exit\_for:

pop ax

pop bx

pop cx ;restoring all register

pop dx

ret

input endp

end main

**UVA 10035 C**

#include<stdio.h>

int main()

{

long int a,b;

int i,r,c,carry;

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

{

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

break;

else

{

r=0;

carry=0;

for(i=1; i<16; i++)

{

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

break;

c=(a%10)+(b%10)+r;

if(c>9)

{

carry++;

r=1;

}

else

{

r=0;

}

a/=10;

b/=10;

}

if(carry==0)

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

else if(carry==1)

printf("1 carry operation.\n");

else

printf("%d carry operations.\n",carry);

}

}

}

**UVA 10055 Assembly**

org 100h

.model small

.stack 100h

.data

number1 db ?

number2 db ?

number3 db ?

hint1 db "enter the two number:$"

hint2 db "the result is: $"

.code

main proc

;loading all data to the data segment

mov ax,@data

mov ds,ax

new:

;displaying the program hint what it is about

mov ah,9

lea dx,hint1

int 21h

;printing a new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;getting user input

mov ah,1

int 21h

;moving the data from al to number1 variable

mov number1,al

;creating a new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;getting the 2nd input

mov ah,1

int 21h

;moving the data from al to number2 variable

mov number2,al

;creating a new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov al,number2 ;moving the number2 value to the al

mov cl,number1 ;moving the number1 value to the cl

cmp cl,al

jg exchange ;if first is big than 2nd then exchange label called

;else continue

mov al,0 ;moving 0 to the al registor

add al,number2 ;add number2 to the al as al=al+number2

sub al,number1 ;then subtract number1 from al that is al=number2-number1

add al,48 ;making the ascii value from the pure value

mov number2,al ;moving the value of al to number2

print:

;initialize the print label

;printing result message

mov ah,9

lea dx,hint2

int 21h

;printing the number2 that means the result

mov dl,number2

mov ah,2

int 21h

;prtinting a new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

loop new ;again re initialize the code

;exchanging the values

;that means number1-number2=result

exchange:

mov al,0

add al,number1

sub al,number2

add al,48

mov number2,al

jmp print ;jumping to the print label

;ending the program

mov ah,4ch

int 21h

main endp

end main

**UVA 10055 C**

#include<stdio.h>

int main()

{

long long int a,b,c;

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

{

if(a>b)

c=a-b;

else

c=b-a;

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

}

return 0;

}

**UVA 10071 Assembly**

org 100h

.model small

.stack 100h

.data

v db "initial velocity : $"

a db "initial acceleration : $"

result db "displacement be in twice of that time : $"

.code

main proc

;storing the data into the data segment

mov ax,@data

mov ds,ax

first:

;hint for velocity

lea dx,v

mov ah,9

int 21h

;getting the first input

mov ah,1

int 21h

sub al,48 ;make the ascii value to the decimal

mov bl,al ;moving the value of al to the variable a as a=al

;checking the input is zero or not

cmp bl,0

je zero ;if zero then jump to the label zero

call printline

;hint for acceleration

lea dx,a

mov ah,9

int 21h

;getting the scnd input

mov ah,1

int 21h

sub al,48 ;make the ascii value to the decimal

;checking the input is zero or not

cmp al,0

je zero ; if zero then jump to the label zero

mov cl,al

call printline

;else it will work

mov al,cl

mul bl ;multiply al by bl and store it in al as we know always result store in al

mov dl,al ;moving the value from al to dl

mov al,2 ;then store the value 2 in al

mul dl ; and then multiply this al=2 by dl result value

mov ah,0

aam ;adjusting after multiply in al and ah that means ax

;making decimal value

add ah,48

add al,48

;moving ax to bx for our register work

mov bx,ax

;hint for result

lea dx,result

mov ah,9

int 21h

;printing the bx value

mov dl,bh

mov ah,2

int 21h

mov dl,bl

int 21h

call printline

jmp again

zero:

;printing zero as output

mov dl,0

mov ah,2

int 21h

;printing new line

mov dl,10

int 21h

mov dl,13

int 21h

;initiating the programme again

again:

jmp first

main endp

proc printline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

ret

printline endp

end main

**UVA 10071 CPP**

#include<bits/stdc++.h>

using namespace std;

int main()

{

int a,b;

while(cin>>a>>b)

{

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

cout<<0<<"\n";

else

cout<<2\*a\*b<<"\n";

}

}

**UVA 10079 Assembly**

.model small

.stack 100h

.data

prog db "Pizza Cutting.$"

about db 10,13,"A negative number terminates the input.$"

value db ?

result db ?

count db ?

.code

main proc

;fetching the data

mov ax,@data

mov ds,ax

;loading the description of the program

lea dx,prog

mov ah,9

int 21h

again: ;reprogrammable label

lea dx,about

mov ah,9

int 21h

call line ;new line

mov value,0 ;making the storing variable value to zero

input1:

;taking input untill new line

mov ah,1

int 21h

mov bl,al

sub bl,48

;comparing for negative number

cmp al,45

je exiting\_input

;comparing for new line

cmp al,13

je result2

;else storing it as 10\*value+bl

mov bh,value

mov al,10

mul bh

add al,bl

mov value,al

jmp input1

result2:

call line

mov result,1 ;taking the result value 1 initially

mov cl,value ;moving input value in the cl registor

mov count,1 ;making the count value 1 as cutted pizza

for:

;changing the result value

mov bh,result

add bh,count

mov result,bh

inc count ;increamenting the count value

;decreament the counter value untill it will appear to zero

dec cl

cmp cl,0

je final ;if zero then we got the final result

jmp for

final:

;making the two digit umber as 16 bit register number in ascii

mov al,1

mov bl,result

mul bl

mov ah,0

aam

;making the ascii to decimal

add al,48

add ah,48

mov bx,ax

;printing the result

mov ah,2

mov dl,bh

int 21h

mov dl,bl

int 21h

jmp again ; initialize the programme again

;when we will get - input then untill new line appear we will take input

exiting\_input:

mov ah,1

int 21h

cmp al,13

je exit

jmp exiting\_input

;label for exiting the function

exit:

mov ah,4ch

int 21h

main endp

;function for a new line

proc line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

ret

line endp

end main

**UVA 10079 C**

#include<stdio.h>

int main()

{

long long int T,sum,i;

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

{

if(T<0)

return 0;

sum=1;

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

sum+=i;

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

}

return 0;

}

**UVA 10170 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

s dw ?

d dw ?

i dw ?

n dw ?

value dw ?

count dw ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov count,0 ;count=0

mov value,0

call input

mov ax,value ;scanf("%d",&s)!=

mov s,ax

printn

mov value,0

call input

mov ax,value ;scanf("%d",&d)!=

mov d,ax

printn

mov n,0 ;n=0;

mov ax,s

mov i,ax ;for(i=s;

for:

mov ax,i

add n,ax ;n+=i;

mov ax,n

cmp ax,d

jle second\_test ;if(n>d) ||

jmp push\_value

second\_test:

mov ax,n

cmp ax,d

je push\_value ;|| n==d)

inc i ;i++)

jmp for

push\_value:

mov ax,i

cmp ax,0 ;checking for i is 0 or not

je pop\_value

xor dx,dx

mov bx,10

div bx ;sum/10

push dx ;pushing last digit as reminder

mov i,ax

inc count ;value length increase

jmp push\_value

pop\_value:

mov ax,count

cmp ax,0 ;checking for value length

je exit2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value

exit2:

printn

jmp again ;calliing the program again

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx

push cx ;saving all data if used

push dx

for\_loop:

mov ah,1

int 21h ;getting input

cbw

cmp ax,13

je exit3 ;checking whether it is new line

cmp ax,32

je exit3 ;checking whether it is space

sub ax,48 ;making pure digit

mov cx,ax ;cx=input

mov ax,value

mov bx,10 ;value=value\*10

mul bx

add ax,cx ;recent result+input

mov value,ax ;value=recent result

jmp for\_loop ;loop call

exit3:

pop dx

pop cx

pop bx ;restoring all registor value

pop ax

ret

input endp

end main

**UVA 10170 C**

#include<stdio.h>

int main()

{

long s,d,i,n;

while(scanf("%ld%ld",&s,&d)==2)

{

n=0;

for(i=s;; i++)

{

n+=i;

if(n>d || n==d)

{

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

break;

}

}

}

return 0;

}

**UVA 10300 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

;int i,j,a,b,c,d,e;

i dw ?

j dw ?

a dw ?

b dw ?

c dw ?

d dw ?

e dw ?

count dw ?

value dw ?

sum dw ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov count,0

mov value,0

call input ;scanf("%d",&a)

mov ax,value

mov a,ax

printn

mov i,0 ;for(i=0;

case\_loop:

mov ax,a

cmp i,ax

jge exit\_case\_loop ;i<a;

mov sum,0 ;int sum=0;

mov value,0

call input

mov ax,value ;scanf("%d",&b);

mov b,ax

printn

mov j,0 ;for(j=0

loop\_for:

mov ax,b

cmp j,ax

jge exit\_loop ;j<b;

mov value,0

call input

mov ax,value ;scanf("%d",&c);

mov c,ax

printn

mov value,0

call input

mov ax,value ;scanf("%d",&d);

mov d,ax

printn

mov value,0

call input

mov ax,value ;scanf("%d",&e);

mov e,ax

printn

mov ax,c

mov bx,e

mul bx ;(c\*e);

add ax,sum

mov sum,ax ; sum=sum+(c\*e);

inc j ;j++)

jmp loop\_for

exit\_loop:

mov ax,sum

cmp ax,0

je print\_num

mov dx,0

mov ax,sum

mov bx,10 ;for printing moving 1 by 1 digit in stack

div bx

mov sum,ax

push dx

inc count

mov cx,count ;counting stack length

jmp exit\_loop

print\_num:

pop dx

add dl,48 ;poping 1 value and printing it

mov ah,2

int 21h

loop print\_num

printn

inc i ;i++)

jmp case\_loop

exit\_case\_loop:

jmp again ;restartting program

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx ;taking all register in stack

push cx

push dx

input\_for:

mov ah,1

int 21h ;scanf

cmp al,13

je exit\_for

sub al,48

cbw ;converting byte to word

mov cx,ax

mov ax,value

mov bx,10 ;saving value

mul bx

add ax,cx

mov value,ax ;value=value\*10 + scanf

jmp input\_for

exit\_for:

pop ax

pop bx

pop cx ;restoring all register

pop dx

ret

input endp

end main

**UVA 10300 C**

#include<stdio.h>

int main()

{

int i,j,a,b,c,d,e;

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

{

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

{

int sum=0;

scanf("%d",&b);

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

{

scanf("%d%d%d",&c,&d,&e);

sum=sum+(c\*e);

}

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

}

}

return 0;

}

**UVA 10302 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a dw ?

b dw ?

c dw ?

i dw ?

value dw ?

count dw ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov count,0 ;count=0

mov value,0

call input

mov ax,value ;scanf("%d",&a)!=

mov a,ax

mov b,ax

printn

mov b,0 ;b=0;

mov i,1 ;for(i=1;

for:

mov ax,i

cmp ax,a

jg push\_value ;i<=a;

mov ax,i

mov bx,i

mul bx

mul bx

mov c,ax ;c=i\*i\*i;

mov ax,c

add b,ax ;b=b+c;

inc i ;i++)

jmp for

push\_value:

mov ax,b

cmp ax,0 ;checking for sum is 0 or not

je pop\_value

xor dx,dx

mov bx,10

div bx ;sum/10

push dx ;pushing last digit as reminder

mov b,ax

inc count ;value length increase

jmp push\_value

pop\_value:

mov ax,count

cmp ax,0 ;checking for value length

je exit2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value

exit2:

printn

jmp again ;calliing the program again

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx

push cx ;saving all data if used

push dx

for\_loop:

mov ah,1

int 21h ;getting input

cbw

cmp ax,13

je exit3 ;checking whether it is new line

cmp ax,32

je exit3 ;checking whether it is space

sub ax,48 ;making pure digit

mov cx,ax ;cx=input

mov ax,value

mov bx,10 ;value=value\*10

mul bx

add ax,cx ;recent result+input

mov value,ax ;value=recent result

jmp for\_loop ;loop call

exit3:

pop dx

pop cx

pop bx ;restoring all registor value

pop ax

ret

input endp

end main

**UVA 10302 C**

#include<stdio.h>

int main()

{

long int a,b,c,i;

while(scanf("%ld",&a)!=EOF)

{

b=0;

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

{

c=i\*i\*i;

b=b+c;

}

printf("%ld\n",b);

}

return 0;

}

**UVA 10327 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a dw ?

b dw 100 dup(0)

c dw ?

i dw ?

j dw ?

d dw ?

value dw ?

count dw ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov count,0 ;count=0

mov value,0

call input

mov ax,value ;scanf("%d",&a)

mov a,ax

printn

mov d,0 ;d=0;

mov i,0 ;for(i=0;

mov si,0

for1:

mov ax,a

cmp i,ax

jge before\_for2 ;i<a;

mov value,0

call input

mov ax,value ;scanf("%d",&b[i])

mov b[si],ax

printn

inc i ;i++)

inc si

inc si ;index increasing

jmp for1

before\_for2:

mov i,1 ;for(i=1;

mov j,0 ;for(j=0;

mov si,0

for2:

mov ax,a

cmp i,ax

jge print ;i<a;

for3:

mov ax,a

dec ax

cmp j,ax

jge increament ;j<a-1;

mov ax,b[si]

cmp ax,b[si+2]

jle increament2 ;if(b[j]>b[j+1])

mov bx,b[si]

mov ax,b[si+2]

mov b[si+2],bx

mov b[si],ax ;swap(b[j] and b[j+1])

inc d ;d++;

increament2:

inc j ;j++)

inc si

inc si ;;index increasing

jmp for3

increament:

inc i ;i++)

mov si,0

mov j,0 ;for(j=0;

jmp for2

print:

mov dx,d

print "Minimum exchange operations : "

cmp dx,0

jg push\_value

add dl,48

mov ah,2 ;if result is zero

int 21h

jmp exit2

push\_value:

mov ax,d

cmp ax,0 ;checking for d is 0 or not

je pop\_value

xor dx,dx

mov bx,10

div bx ;d/10

push dx ;pushing last digit as reminder

mov d,ax

inc count ;value length increase

jmp push\_value

pop\_value:

mov ax,count

cmp ax,0 ;checking for value length

je exit2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value

exit2:

printn

jmp again ;calliing the program again

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx

push cx ;saving all data if used

push dx

for\_loop:

mov ah,1

int 21h ;getting input

cbw

cmp ax,13

je exit3 ;checking whether it is new line

cmp ax,32

je exit3 ;checking whether it is space

sub ax,48 ;making pure digit

mov cx,ax ;cx=input

mov ax,value

mov bx,10 ;value=value\*10

mul bx

add ax,cx ;recent result+input

mov value,ax ;value=recent result

jmp for\_loop ;loop call

exit3:

pop dx

pop cx

pop bx ;restoring all registor value

pop ax

ret

input endp

end main

**UVA 10327 C**

#include<stdio.h>

int main()

{

int a,b[1000],c,i,j,d;

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

{

d=0;

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

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

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

for(j=0; j<a-1; j++)

{

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

{

c=b[j];

b[j]=b[j+1];

b[j+1]=c;

d++;

}

}

printf("Minimum exchange operations : %d\n",d);

}

return 0;

}

**UVA 10346 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a dw ?

b dw ?

x dw ?

z dw ?

n dw ?

sum dw ?

value dw ?

count dw ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

again:

mov count,0 ;count=0

mov value,0

call input

mov ax,value ;scanf("%d",&a)!=

mov a,ax

mov value,0

call input

mov ax,value ;scanf("%d",&b)!=

mov b,ax

printn

cmp ax,1

jle exit ;if(b>1) then exit

mov ax,a

mov n,ax ;n=a

mov sum,0 ;sum=0

while:

mov ax,a

cmp ax,b

jl print\_sum ;while(a>=b)

xor dx,dx

mov ax,a

mov bx,b ;a/b

div bx

mov z,ax ;z=a/b;

mov x,dx ;x=a%b;

add ax,dx

mov a,ax ;a=z+x;

mov ax,z

add sum,ax ;sum=sum+z;

jmp while

print\_sum:

mov ax,n

add sum,ax ;sum=sum+n

push\_value:

mov ax,sum

cmp ax,0 ;checking for sum is 0 or not

je pop\_value

xor dx,dx

mov bx,10

div bx ;sum/10

push dx ;pushing last digit as reminder

mov sum,ax

inc count ;value length increase

jmp push\_value

pop\_value:

mov ax,count

cmp ax,0 ;checking for value length

je exit2

dec count

pop dx

add dx,48

mov ah,2 ;printing digit from stack

int 21h

jmp pop\_value

exit2:

jmp again ;calliing the program again

exit:

mov ah,4ch

int 21h

main endp

input proc

push ax

push bx

push cx ;saving all data if used

push dx

for\_loop:

mov ah,1

int 21h ;getting input

cbw

cmp ax,13

je exit3 ;checking whether it is new line

cmp ax,32

je exit3 ;checking whether it is space

sub ax,48 ;making pure digit

mov cx,ax ;cx=input

mov ax,value

mov bx,10 ;value=value\*10

mul bx

add ax,cx ;recent result+input

mov value,ax ;value=recent result

jmp for\_loop ;loop call

exit3:

pop dx

pop cx

pop bx ;restoring all registor value

pop ax

ret

input endp

end main

**UVA 10346 C**

#include<stdio.h>

int main()

{

int a,b,x,z;

while(scanf("%d%d",&a,&b)!=EOF && b>1)

{

int n=a, sum=0;

while(a>=b){

z=a/b;

x=a%b;

a=z+x;

sum=sum+z;

}

printf("%d\n",sum+n);

}

return 0;

}

**UVA 10469 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a db ?

b db ?

.code

main proc

again:

mov dl,0

call input ;taking first input

mov a,dl

printn

mov dl,0

call input ;taking second input

mov b,dl

printn

mov bl,a

mov bh,b

xor bh,bl ;first ^ second

cmp bh,9

jg greater ;if(output>9)

;print 2 digit

mov dl,bh

add dl,48 ;else

mov ah,2 ;print 1 digit

int 21h

jmp restart

greater:

mov al,bh

xor ah,ah

aaa

mov bx,ax

mov dl,bh

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bl

add dl,48 ;print low digit

mov ah,2

int 21h

restart:

printn

jmp again ;restart the program

main endp

input proc

input\_loop:

mov ah,1

int 21h

cmp al,13

je return

sub al,48

mov bl,al

mov al,dl

mov bh,10

mul bh

add al,bl

mov dl,al

jmp input\_loop

return:

ret

input endp

end main

UVA 10469 C

#include <iostream>

using namespace std;

int main()

{

int i, j;

while (cin >> i >> j)

cout << (i ^ j) << '\n';

}

**UVA 10499 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

.code

main proc

input:

mov ah,1

int 21h ;scanf("%ld",&a)

cmp al,45

je finish ;if(a>0) then finish

cmp al,1

je zero\_output ;if(a==1) then zero\_output

sub al,48

mov bl,25

mul bl

mov ah,0

mov bl,10 ;25\*a

div bl

mov bx,ax

printn

mov ah,2

mov dl,bl ;as it is 2 digit value then prin high value which is stored in bl register

add dl,48

int 21h

mov dl,bh

add dl,48 ;low value which is in bh

int 21h

printn "%"

jmp input

zero\_output:

printn "0%" ;printf("0%%\n");

jmp input

finish:

mov ah,1 ;taking another any input after - sign

int 21h

printn

mov ah,4ch ;exit

int 21h

main endp

end main

**UVA 10499 C**

#include<stdio.h>

int main()

{

long int a;

while(scanf("%ld",&a)==1 && a>0)

{

if(a==1)

printf("0%%\n");

else

printf("%ld%%\n",25\*a);

}

return 0;

}

**UVA 10970 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a db ?

b db ?

.code

main proc

again:

mov dl,0

call input ;taking first input

mov a,dl

printn

mov dl,0

call input ;taking second input

mov b,dl

printn

mov ah,0

mov al,a

mov bl,b

mul bl

sub al,1

cmp al,9

jg greater

mov dl,al

add dl,48 ;else

mov ah,2 ;print 1 digit

int 21h

jmp restart

greater:

xor ah,ah

aaa

mov bx,ax

mov dl,bh

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bl

add dl,48 ;print low digit

mov ah,2

int 21h

restart:

printn

jmp again ;restart the program

main endp

input proc

input\_loop:

mov ah,1

int 21h

cmp al,13

je return

sub al,48

mov bl,al

mov al,dl

mov bh,10

mul bh

add al,bl

mov dl,al

jmp input\_loop

return:

ret

input endp

end main

**UVA 10970 C**

#include<stdio.h>

int main()

{

int a,b,sum;

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

{

if(1<=a<=300 && 1<=b<=300)

{

sum=n\*m-1;

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

}

}

return 0;

}

**UVA 11150 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

n db ?

sum db ?

.code

main proc

again:

mov dl,0 ;scanf("%d", &n)

call input

cmp dl,0 ;compare a!=0

je exit

mov n,dl

mov sum,dl ;sum = 0;

printn

while:

mov bh,n

cmp bh,3 ;while(n > 0)

jl checking

mov ah,0

mov al,n

mov bh,3 ;sum += n/3;

div bh

mov bh,sum

add bh,al

mov sum,bh

add ah,al ;n = n/3 + n%3;

mov n,ah

jmp while

checking:

mov ah,n

cmp ah,2

je add\_sum ;if(n==2) then add\_sum

jmp print

add\_sum:

inc sum ;n++;

print:

mov al,sum

cmp al,15

jg greater\_than\_15

cmp al,9

jg greater\_than\_9

mov dl,sum

add dl,48 ;else

mov ah,2 ;print 1 digit

int 21h

jmp restart

greater\_than\_15:

mov ah,0

mov al,sum

mov bl,10

div bl

mov bx,ax

mov dl,bl

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bh

add dl,48 ;print low digit

mov ah,2

int 21h

jmp restart

greater\_than\_9:

mov al,sum

xor ah,ah ;printf("%d\n", sum);

aaa ;;print 2 digit

mov bx,ax

mov dl,bh

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bl

add dl,48 ;print low digit

mov ah,2

int 21h

restart:

printn

jmp again ;restart the program

exit:

mov ah,4ch

int 21h

main endp

input proc

input\_loop:

mov ah,1

int 21h

cmp al,13

je return

sub al,48

mov bl,al ;taking multiple value digit

mov al,dl

mov bh,10

mul bh

add al,bl

mov dl,al

jmp input\_loop

return:

ret

input endp

end main

**UVA 11150 C**

#include <stdio.h>

int main() {

int n;

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

int sum = n;

while(n >= 3) {

sum += n/3;

n = n/3 + n%3;

}

if(n == 2) sum++;

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

}

return 0;

}

**UVA 11172 Assembly**

org 100h

.model small

.stack 100h

.data

a db "enter two digit number or 1 digit along 0 in the first case",10,13,"such as 12(two) or 01(one)$"

b db 10,13,"for exit input two 0(zero)$"

c db "input number 2: $"

d db 10

e db 10,13,"input number 1: $"

.code

main proc

;loading data into the data segment

mov ax,@data

mov ds,ax

;printing the hint message

lea dx,a

mov ah,9

int 21h

again:

;the label for re starting the code

;hint for exiting the code

lea dx,b

mov ah,9

int 21h

;input message for input 1

lea dx,e

mov ah,9

int 21h

;input frst digit of 2digits number

mov ah,1

int 21h

sub al,48

mov a,al ;moving the frst value to the a variable

;input the scnd number of 2digits number

mov ah,1

int 21h

sub al,48

mov b,al ;moving the scnd value to the b variable

;newline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;hint for scnd two digit number

lea dx,c

mov ah,9

int 21h

mov al,a ;moving the frst digit to al

mul d ;multiplicate the value of al by d=10

mov ah,0 ;vacant the ah

aam ;adjusting after multiplication the ax

add al,b ;then adding the al with b which will compatible in ax

;getting the ascii value

add ah,48

add al,48

mov bx,ax ;moving the ax to bx

;checking if the input is zero

cmp bh,48

je zero1

zero1:

cmp bl,48

je exit ;if ax==00 then exit

;else continue

;input frst digit of 2digits number

mov ah,1

int 21h

sub al,48

mov a,al ;moving the frst value to the a variable

;input the scnd number of 2digits number

mov ah,1

int 21h

sub al,48

mov b,al ;moving the scnd value to the b variable

;newline

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov al,a ;moving the frst digit to al

mul d ;multiplicate the value of al by d=10

mov ah,0 ;vacant the ah

aam ;adjusting after multiplication the ax

add al,b ;then adding the al with b which will compatible in ax

;getting the ascii value

add ah,48

add al,48

mov cx,ax ;moving the ax to cx

;comparing either 2 high byte is equal

cmp bh,ch

je equal

;comparing either frst high is big than 2nd

cmp bh,ch

jg greater

jmp less

equal:

;comparing either 2 low byte is equal

cmp bl,cl

je equal2

;comparing either last low is big than 2nd

cmp bl,cl

jg greater

jmp less

equal2:

;printing the equal message

mov dl,"="

mov ah,2

int 21h

jmp newline

greater:

;printing the greater message

mov dl,">"

mov ah,2

int 21h

jmp newline

less:

;printing the lesser message

mov dl,"<"

mov ah,2

int 21h

jmp newline

;newline

newline:

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

jmp again

exit:

;exiting the program

mov ah,4ch

int 21h

main endp

end main

**UVA 11172 C**

#include<stdio.h>

int main()

{

int a,b,c,i;

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

{

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

{

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

if(b>c)

printf(">\n");

else if(b<c)

printf("<\n");

else

printf("=\n");

}

}

return 0;

}

**UVA 11479 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

a db ?

b db ?

c db ?

case db 49

.code

main proc

;fetching data from data segment

mov ax,@data

mov ds,ax

;getting the test case

mov ah,1

int 21h

sub al,48

mov cl,al ;count=testcase

mov ch,0

printn ;\n

all:

mov ah,1

int 21h

sub al,48 ;taking input a

mov a,al

print " "

int 21h

sub al,48 ;taking input b

mov b,al

print " "

int 21h

sub al,48 ;taking input c

mov c,al

printn

mov bl,a

add bl,b ;if((a+b)<=c))

cmp bl,c

jl invalid ;jump invalid

mov bl,b

add bl,c

cmp bl,a ;if((c+b)<=a))

jl invalid ;jump invalid

mov bl,a

add bl,c

cmp bl,b ;if((a+c)<=b))

jl invalid ;jump invalid

mov bl,a

cmp bl,0 ;if(a==0)

jl invalid ;jump invalid

mov bl,a

cmp bl,0 ;if(b==0)

jl invalid ;jump invalid

mov bl,a

cmp bl,0 ;if(c==0)

jl invalid ;jump invalid

mov bl,a

cmp bl,b ;if(a==b)

je andtrue ;jump and true

mov bl,b

cmp bl,c

je Isosceles

mov bl,a

cmp bl,c

je Isosceles

;printf("Case %ld: Scalene\n",i);

print "Case "

mov ah,2

mov dl,case

int 21h

printn ": Scalene"

jmp again\_start

andtrue:

mov bl,b

cmp bl,c ;if(a==b) && (b==c)

je Equilateral ;jump equilateral

;printf("Case %ld: Isosceles\n",i);

Isosceles:

print "Case "

mov ah,2

mov dl,case

int 21h

printn ": Isosceles"

jmp again\_start

;printf("Case %ld: Equilateral\n",i);

Equilateral:

print "Case "

mov ah,2

mov dl,case

int 21h

printn ": Equilateral"

jmp again\_start

;printf("Case %ld: Invalid\n",i);

Invalid:

print "Case "

mov ah,2

mov dl,case

int 21h

printn ": Invalid"

again\_start:

mov al,case

inc al ;case++

mov case,al

loop all ;testcase++

;exiting the program

exit:

mov ah,4ch

int 21h

main endp

end main

**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;

}

**UVA 11498 Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data

;variable and data section

testcase db ?

first db ?

scnd db ?

test1 db ?

test2 db ?

;string for declaring the result

divisa db "divisa",10,13,"$"

ne db "NE",10,13,"$"

se db "SE",10,13,"$"

no db "NO",10,13,"$"

so db "SO",10,13,"$"

.code

;code section

main proc

;main procedure start

mov ax,@data

mov ds,ax

; |

; |

; NO | NE

; |

; --------divisa---------

; |

; SO | SE

; |

; |

;label for restarting the program

again:

xor dl,dl ;making the dl registor value initially zero

call input ;calling the input fuction for taking input

mov testcase,dl

;comparing the input test case if zero

cmp testcase,0

je exit ;if zero then the program will terminate

printn ;printing a new line

;again taking the input and moving it to the first variable

xor dl,dl

call input

mov first,dl

printn

;taking the input and moving it to the scnd variable

xor dl,dl

call input

mov scnd,dl

mov cl,testcase ;taking the testcase to the counter register for making the loop length

printn

;now taking the two number for testcase length to check

testing\_input:

;checking the counter value whether zero or not

cmp cl,0

je last ;if zero then terminate to the restarting program

dec cl

;taking the first input for divisia checking

xor dl,dl

call input

mov test1,dl

printn

;taking the scnd input

xor dl,dl

call input

mov test2,dl

;moving to the another register for checking

mov bh,test1

mov bl,test2

;comparing the first input to the first fixed divisia value

cmp bh,first

jg first\_greater ;if greater then go to label first greater

jl first\_less ;else go to the first\_less label

;if all above are wrong then given and exist must be equal

equal:

printn

;if equal then priting the divisia string

lea dx,divisa

mov ah,9

int 21h

loop testing\_input

first\_greater:

cmp bl,scnd

jg firstgreater\_scndgreater ;if bl>scnd then jump to the label firstgreater\_scndgreater

jl firstgreater\_scndless ;else to the label firstgreater\_scndless

jmp equal ;if all above are wrong then it must be equal

first\_less:

cmp bl,scnd

jg firstless\_scndgreater ;if bl>scnd then jump to the label firstless\_scndgreater

jl firstless\_scndless ;;else to the label firstless\_scndless

jmp equal ;if all above are wrong then it must be equal

firstgreater\_scndgreater:

printn

;if bh>first and bl>scnd then it will print NE

lea dx,ne

mov ah,9

int 21h

loop testing\_input

firstgreater\_scndless:

printn

;if bh>first and bl<scnd then it will print NE

lea dx,se

mov ah,9

int 21h

jmp last

firstless\_scndgreater:

printn

;if bh<first and bl>scnd then it will print NE

lea dx,no

mov ah,9

int 21h

loop testing\_input

firstless\_scndless:

printn

;if bh<first and bl<scnd then it will print NE

lea dx,so

mov ah,9

int 21h

loop testing\_input

last:

;when loop will end then the last label will be called

jmp again

;the exiting label

exit:

mov ah,4ch

int 21h

main endp

;procedure for taking input

input proc

for:

mov ah,1

int 21h

cmp al,13 ;if input is new line the it will refused to take input

je exit\_for

sub al,48 ;taking the ascii value

mov bh,10 ;bh=10

mov bl,al ;bl=input value al

mov al,dl ;stored value dl to the al

mul bh ;al=al\*bh

add al,bl ;al=al+bl

mov dl,al ;dl=al final

jmp for ;again taking input

;when exit for called then it will return the ascii value store in dl

exit\_for:

ret

input endp

end main

**UVA 11498 C**

#include<stdio.h>

int main()

{

int t,m,n,x,y;

while(scanf("%d",&t)==1 && t!=0)

{

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

while(t-->0)

{

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

if(x==m || y==n)

printf("divisa\n");

else if(x>m && y>n)

printf("NE\n");

else if(x>m && y<n)

printf("SE\n");

else if(x<m && y>n)

printf("NO\n");

else if(x<m && y<n)

printf("SO\n");

}

}

return 0;

}

**UVA 11877 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

n db ?

sum db ?

.code

main proc

again:

mov dl,0 ;scanf("%d", &n)

call input

cmp dl,0 ;compare a!=0

je exit

mov n,dl

mov sum,0 ;sum = 0;

printn

while:

mov bh,n

cmp bh,0 ;while(n > 0)

jle print

mov ah,0

mov al,n

mov bh,3 ;sum += n/3;

div bh

mov bh,sum

add bh,al

mov sum,bh

add ah,al ;n = n/3 + n%3;

mov n,ah

cmp ah,2

je checking ;if(a==2 ||

cmp ah,1

je checking ;|| a==1) then jump checking

jmp while

checking:

cmp ah,2

je add\_n ;if(a==2) then add\_n

jmp print

add\_n:

inc n ;n++;

jmp while

print:

mov al,sum

cmp al,15

jg greater\_than\_15

cmp al,9

jg greater\_than\_9

mov dl,sum

add dl,48 ;else

mov ah,2 ;print 1 digit

int 21h

jmp restart

greater\_than\_15:

mov ah,0

mov al,sum

mov bl,10

div bl

mov bx,ax

mov dl,bl

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bh

add dl,48 ;print low digit

mov ah,2

int 21h

jmp restart

greater\_than\_9:

mov al,sum

xor ah,ah ;printf("%d\n", sum);

aaa ;;print 2 digit

mov bx,ax

mov dl,bh

add dl,48 ;print high digit

mov ah,2

int 21h

mov dl,bl

add dl,48 ;print low digit

mov ah,2

int 21h

restart:

printn

jmp again ;restart the program

exit:

mov ah,4ch

int 21h

main endp

input proc

input\_loop:

mov ah,1

int 21h

cmp al,13

je return

sub al,48

mov bl,al ;taking multiple value digit

mov al,dl

mov bh,10

mul bh

add al,bl

mov dl,al

jmp input\_loop

return:

ret

input endp

end main

**UVA 11877 C**

#include<stdio.h>

int main()

{

int a,b,c,sum;

while(scanf("%d",&a)==1 && a!=0)

{

sum=0;

while(a>0){

b=a/3;

c=a%3;

a=b+c;

sum=sum+b;

if(a==2 || a==1)

{

if(a==2)

a+=1;

else

break;

}

}

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

}

return 0;

}

**UVAS 12646 Assembly**

include 'emu8086.inc' ;importing a header file

.model small

.stack 100h

.data

;variable declaration

a db ?

b db ?

c db ?

.code

main proc

;fetching data into the data segment

mov ax,@data

mov ds,ax

;restarting the program

;outer while(true)

again:

mov bl,1 ;for(int i=1)

;for loop start

input\_for:

cmp bl,3 ;checking i<=3?

jg operation

mov ah,1 ;scanf

int 21h

sub al,48

cmp bl,1

je invoke\_a ;if(bl==1) then jumping tothe label invoke\_a

cmp bl,2

je invoke\_b ;if(bl==2) then jumping tothe label invoke\_b

mov c,al ;c=al

inc bl

jmp input\_for ;again for loop start with bl++

invoke\_a:

mov a,al ;a=al

inc bl

jmp input\_for ;again for loop start with bl++

invoke\_b:

mov b,al ;b=al

inc bl

jmp input\_for ;again for loop start with bl++

;after loop end ooperation will start

operation:

printn ;new line

mov bl,a

cmp bl,0

je a\_zero ;if a==0

jmp a\_one ;else a==1

a\_zero:

mov bl,b

cmp bl,0

je a\_zero\_b\_zero ;if a==0 and b==0

jmp a\_zero\_b\_one ;else a==0 and b==1

a\_one:

mov bl,b

cmp bl,0

je a\_one\_b\_zero ;if a==1 and b==0

jmp a\_one\_b\_one ;else a==1 and b==1

a\_zero\_b\_one:

mov bl,c

cmp bl,0

je a\_zero\_b\_one\_c\_zero ;if a==0 and b==1 and c==0

jmp a\_zero\_b\_one\_c\_one ;else a==0 and b==1 and c==1

a\_zero\_b\_zero:

mov bl,c

cmp bl,1

je a\_zero\_b\_zero\_c\_one ;if a==0 and b==0 and c==1

jmp last ;else jump last

a\_one\_b\_zero:

mov bl,c

cmp bl,0

je a\_one\_b\_zero\_c\_zero ;if a==1 and b==0 and c==0

jmp a\_one\_b\_zero\_c\_one ;else a==1 and b==0 and c==1

a\_one\_b\_one:

mov bl,c

cmp bl,0

je a\_one\_b\_one\_c\_zero ;if a==1 and b==1 and c==0

jmp last ;else jump last

a\_zero\_b\_zero\_c\_one:

printn "C" ;printf "C"

jmp repro

a\_zero\_b\_one\_c\_zero:

printn "B" ;printf "B"

jmp repro

a\_zero\_b\_one\_c\_one:

printn "A" ;printf "A"

jmp repro

a\_one\_b\_zero\_c\_zero:

printn "A" ;printf "A"

jmp repro

a\_one\_b\_zero\_c\_one:

printn "B" ;printf "B"

jmp repro

a\_one\_b\_one\_c\_zero:

printn "C" ;printf "C"

jmp repro

last:

printn "\*" ;printf "\*"

;starting again that means while loop continue

repro:

jmp again

main endp

end main

**UVA 12646 C**

#include<stdio.h>

int main()

{

int a,b,c;

while(scanf("%d%d%d",&a,&b,&c)==3 && (a==0 || a==1 || b==0 || b==1 || c==0 || c==1))

{

if((a==0 && b==0 && c==1) || (a==1 && b==1 && c==0))

printf("C\n");

else if((a==0 && b==1 && c==0) || (a==1 && b==0 && c==1))

printf("B\n");

else if((a==1 && b==0 && c==0) || (a==0 && b==1 && c==1))

printf("A\n");

else

printf("\*\n");

}

return 0;

}

**UVA 12700 Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data

;making all the variable

testout db ?

b db ?

w db ?

t db ?

a db ?

i db ?

j db ?

testin db ?

casestring db "Case $"

abandoned db ": ABANDONED$"

whitewash db ": WHITEWASH$"

banglawash db ": BANGLAWASH$"

draw db ": DRAW $"

bangladesh db ": BANGLADESH $"

www db ": WWW $"

case db ?

about db "These letters will be either `B' or `W' or `T' or `A'.",10,13,"$"

about\_2 db "here B=Bangladesh W=WWW , T=Tie and A=Abandoned",10,13,"$"

.code

main proc

;fetching the data variable

mov ax,@data

mov ds,ax

;about string output

lea dx,about

mov ah,9

int 21h

lea dx,about\_2

int 21h

;taking the testcase input

xor dl,dl

call input ;scanf called(function)

mov testout,dl ;testout=scanf(dl)

mov i,0 ;int i=0

mov case,49 ;int case=1(49 in ascii)

;initial outer for loop

testcaseout:

printn ;new line

;i<=testout

mov bl,testout

cmp i,bl

je exit

inc i ;i++

;initializing all variable 0

mov b,0

mov w,0

mov t,0

mov a,0

;scanf

xor dl,dl

call input

printn ;new line

mov testin,dl ;testin=scanf(dl)

mov j,0 ;int j=0

;for loop 2

testcasein:

;checking j<=testin

mov bl,testin

cmp j,bl

je next\_phase ;after loop ending ->next phase

inc j ;j++

;getting another input

mov ah,1

int 21h

;dl=got input

mov dl,al

;checking the input either =A

cmp dl,65

je aplus

;checking the input either =B

cmp dl,66

je bplus

;checking the input either =T

cmp dl,84

je tplus

;checking the input either =W

cmp dl,87

je wplus

;if all are false then jumping testcasein for taking again input

jmp testcasein

;label for a++

aplus:

inc a

jmp testcasein

;label for b++

bplus:

inc b

jmp testcasein

;label for t++

tplus:

inc t

jmp testcasein

;label for w++

wplus:

inc w

jmp testcasein

next\_phase:

mov bl,b

cmp bl,0

je b\_zero ;checking if b==0

;else checking if w==0

mov bl,w

cmp w,0

je w\_zero

jmp others ;or to the others label

;if(b==0)

b\_zero:

mov bl,w

cmp bl,0

je b\_zero\_w\_zero ;if b==0 and w==0

mov bl,t

cmp bl,0

je whitewash\_final ;if(b==0 and t==0)

;then whitewash label

jmp others

w\_zero:

mov bl,t

cmp bl,0

je banglawash\_final ;if(w==0 and t==0)

;then banglawash label

jmp others

b\_zero\_w\_zero:

mov bl,testin

cmp bl,a

je abandoned\_final ;if(b==0 and w==0 and a==testin)

;then abandoned label

others:

mov bl,b

cmp bl,w

je draw\_final ;if(b==w)

;then draw final label

cmp bl,w

jg bangladesh\_final ;else if(b>w)

;then bangladesh final label

jmp www\_final ;else

;then www final

;printing the whitewash string

whitewash\_final:

printn

call casing

lea dx,whitewash

mov ah,9

int 21h

jmp last

;printing the banglawash string

banglawash\_final:

printn

call casing

lea dx,banglawash

mov ah,9

int 21h

jmp last

;printing the abandoned string

abandoned\_final:

printn

call casing

lea dx,abandoned

mov ah,9

int 21h

jmp last

;printing the draw with score string

draw\_final:

printn

call casing

lea dx,draw

mov ah,9

int 21h

mov ah,2

mov bl,b

add bl,48

mov dl,bl

int 21h

mov dl,"-"

int 21h

mov bl,t

add bl,48

mov dl,bl

int 21h

jmp last

;printing the bangladesh\_final with score string

bangladesh\_final:

printn

call casing

lea dx,bangladesh

mov ah,9

int 21h

mov ah,2

mov bl,b

add bl,48

mov dl,bl

int 21h

mov dl,"-"

int 21h

mov bl,w

add bl,48

mov dl,bl

int 21h

jmp last

;printing the www\_final with score string

www\_final:

printn

call casing

lea dx,www

mov ah,9

int 21h

mov ah,2

mov bl,w

add bl,48

mov dl,bl

int 21h

mov dl,"-"

int 21h

mov bl,b

add bl,48

mov dl,bl

int 21h

last:

jmp testcaseout

;exit

exit:

mov ah,4ch

int 21h

main endp

;procedure for printing the case string with number

casing proc

lea dx,casestring

mov ah,9

int 21h

mov dl,case

mov ah,2

int 21h

inc case

ret

casing endp

;procedure for taking input

input proc

for:

mov ah,1

int 21h

cmp al,13 ;if input is new line the it will refused to take input

je exit\_for

sub al,48 ;taking the ascii value

mov bh,10 ;bh=10

mov bl,al ;bl=input value al

mov al,dl ;stored value dl to the al

mul bh ;al=al\*bh

add al,bl ;al=al+bl

mov dl,al ;dl=al final

jmp for ;again taking input

;when exit for called then it will return the ascii value store in dl

exit\_for:

ret ;returning the input value

input endp

end main

**UVA 12700 C**

#include<stdio.h>

int main()

{

int ti,i,m,b,w,a,t,cas=1;

char n;

scanf("%d",&ti);

while(ti-->0)

{

b=0;

w=0;

t=0;

a=0;

scanf("%d",&m);

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

{

scanf("%c",&n);

switch (n)

{

case 'B':

b++;

break;

case 'W':

w++;

break;

case 'T':

t++;

break;

case 'A':

a++;

break;

}

}

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

printf("Case %d: ABANDONED\n",cas++);

else if(w==0 && t==0)

printf("Case %d: BANGLAWASH\n",cas++);

else if(b==0 && t==0)

printf("Case %d: WHITEWASH\n",cas++);

else if(b>w)

printf("Case %d: BANGLADESH %d - %d\n",cas++,b,w);

else if(b<w)

printf("Case %d: WWW %d - %d\n",cas++,w,b);

else if(b==w)

printf("Case %d: DRAW %d %d\n",cas++,b,t);

}

return 0;

}

**UVA 12917 Assembly**

include 'emu8086.inc'

.model small

.stack 100h

.data

hint db "`Props win!' if the props survive, otherwise print `Hunters win!'.$"

hunters db "Hunters win!$"

props db "Props win!$"

a db ?

b db ?

c db ?

.code

main proc

;fetching the data

mov ax,@data

mov ds,ax

;printing the hint data

lea dx,hint

mov ah,9

int 21h

printn

;restarting the program again

again:

;getting the first input

xor dl,dl

call input

mov a,dl

printn

;getting the second input

xor dl,dl

call input

mov b,dl

printn

;getting the third input

xor dl,dl

call input

mov c,dl

printn

;moving the 3 data to the registor

mov bl,a

mov bh,b

mov cl,c

;adding the first 2 data

add bl,bh

;comparing the adding the data to the 3rd data

cmp bl,cl

jg hunter ;if 1+2>3 then jump to the hunter label

;else

;printing the props wining string

lea dx,props

mov ah,9

int 21h

printn

jmp again ;jumping to the again label

hunter:

;printing the props wining string

lea dx,hunters

mov ah,9

int 21h

printn

jmp again

main endp

;procedure for taking input

input proc

for:

mov ah,1

int 21h

cmp al,13 ;if input is new line the it will refused to take input

je exit\_for

sub al,48 ;taking the ascii value

mov bh,10 ;bh=10

mov bl,al ;bl=input value al

mov al,dl ;stored value dl to the al

mul bh ;al=al\*bh

add al,bl ;al=al+bl

mov dl,al ;dl=al final

jmp for ;again taking input

;when exit for called then it will return the ascii value store in dl

exit\_for:

ret ;returning the input value

input endp

end main

**UVA 12917 C**

#include<stdio.h>

int main()

{

int p,h,o;

while(scanf("%d%d%d",&p,&h,&o)==3)

{

if(p+h>o)

printf("Hunters win!\n");

else

printf("Props win!\n");

}

return 0;

}

**UVA 12952 Assembly**

org 100h

.model small

.stack 100h

.data

about db "a program to determine the value of the third card",10,13,"that maximizes the probability of that ",10,13,"player winning the game.$"

frst db 10,13,"Enter the frst number A (1<A<13): ",10,13,"$"

scnd db 10,13,"Enter the second number B (1<B<13): ",10,13,"$"

win db "Winning Card: ",10,13,"$"

.code

main proc

;loading data to the data segment

mov ax,@data

mov ds,ax

;loading the about

lea dx,about

mov ah,9

int 21h

again\_start:

;setting message for first input

lea dx,frst

mov ah,9

int 21h

;getting the first input

mov ah,1

int 21h

mov bh,al

;setting message for second input

lea dx,scnd

mov ah,9

int 21h

;getting the second input

mov ah,1

int 21h

mov bl,al

;new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

;output message

lea dx,win

mov ah,9

int 21h

;checking what is greater than other

cmp bl,bh

jge greater ;if bl greater than and equal with bh then jump greater label

jmp lesser ;else lesser label

greater:

;printing greater or equal as bl

mov dl,bl

mov ah,2

int 21h

;procedure start again for next test case

jmp again\_start

lesser:

;printing leasser as bh

mov dl,bh

mov ah,2

int 21h

;procedure start again for next test case

jmp again\_start

main endp

end main

**UVA 12952 C**

#include<stdio.h>

int main()

{

int a,b;

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

{

if(a==b)

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

else if(a>b)

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

else

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

}

return 0;

}

**UVA 12992 Assembly**

org 100h

.model small

.stack 100h

.data

;defining the hint mesagge

about db "what's the minimal number of bottles needed",10,13,"if he want to bring N types of medicine.$"

test\_case db 10,13,"give the test case number:",10,13,"$"

nth\_value db 10,13,"give the desired n'th value: ",10,13,"$"

minimal\_number db 10,13,"the minimal number of bottles Huatuo needed: ",10,13,"$"

case\_print db 10,13,"Case %$"

case db ?

sum db ?

i db ?

.code

main proc

;load the data value to the data segment

mov ax,@data

mov ds,ax

;load the about of the programme

lea dx,about

mov ah,9

int 21h

;loading the program again

again\_programme:

;showing the message for getting the input test case

lea dx,test\_case

mov ah,9

int 21h

;getting the input

mov ah,1

int 21h

mov case,al

mov i,49 ;moving case value as 1

;making the value of bx zero

mov bh,0

mov bl,0

input:

;showing the message for getting the nth value case

lea dx,nth\_value

mov ah,9

int 21h

;getting the input which nth value we need

mov ah,1

int 21h

mov cl,al

mov sum,1 ;making the value of sum initially 1

getting\_sum:

dec cl ;for calculating nth value decreasing it

cmp cl,48 ;comparing either the nth is zero or not

je print ;if true then print the sum

;checking either the value of sum is greater than 9 or not

mov ch,sum

cmp ch,9

jg greater\_nine ;if true then only increament the lower part of bx as bl

;else increasing the sum value

;increamenting value of sum by 2 in every cases

inc ch

inc ch

mov sum,ch

;storing the sum value in ax registor not bothering about 2 digits

mov al,0

add al,sum

mov ah,0

aaa

add ah,48 ;making ascii the higher part

add al,48 ;making ascii the higher part

mov bx,ax ;storing the ax value to the bx

jmp getting\_sum ;again checking the loop getting\_sum

greater\_nine: ;if greater nine then increament only lower part

inc bl ;once

inc bl ;twice

jmp getting\_sum ;again checking the loop getting\_sum

print:

;printing the case message

mov ah,9

lea dx,case\_print

int 21h

;printing the case number

mov ah,2

mov dl,i

int 21h

mov dl,":"

int 21h

cmp sum,9

jle print\_less

;priniting the sum value from bx registor

mov ah,2

mov dl,bh

int 21h

mov dl,bl

int 21h

inc\_case:

;increamenting the case value

mov bh,i

inc bh

mov i,bh

;if case value is lesser than input test case then again get input

cmp bh,case

jle input

;else programme restarting

jmp again\_programme

print\_less:

;printing the value less or equal 9

add sum,48

mov ah,2

mov dl,sum

int 21h

jmp inc\_case ;jumping to the increment case

;programme exit

exit:

mov ah,4ch

int 21h

main endp

end main

**UVA 12992 C**

#include<stdio.h>

int main()

{

int t,sum,n,i,cas=1;

scanf("%d",&t);

while(t-->0)

{

scanf("%d",&n);

i=1;

sum=1;

while(i!=n)

{

sum+=2;

i++;

}

printf("Case #%d: %d\n",cas++,sum);

}

return 0;

}

**UVA 13012 Assembly**

org 100h

.model small

.stack 100h

.data

about db ":it is about priniting how many number",10,13,"are containing in below 5 extra answer:(for 1 digit 1 only)",10,13,"$"

correct\_ans db 10,13,"enter the correct answer:(with 1 digit)",10,13,"$"

student\_ans db "give the 5 ans in 1 digit with space or not:",10,13,"$"

how\_correct db 10,13,"the correct ans are: ",10,13,"$"

.code

main proc

mov ax,@data

mov ds,ax

lea dx,about

mov ah,9

int 21h

program\_start:

;correct ans input hint

lea dx,correct\_ans

mov ah,9

int 21h

;taking the first correct answer input

mov ah,1

int 21h

mov bh,al

;printing a new line

mov ah,2

mov dl,10

int 21h

mov dl,13

int 21h

mov bl,48 ;moving the flag value how many are correct value 0 as 48 ascii

mov cl,48 ;moving the loop length value 0 as 48 ascii

;Student 5 ans input hint

lea dx,correct\_ans

mov ah,9

int 21h

input:

;taking the input

mov ah,1

int 21h

;space checking as ascii value of space is 32

cmp al,32

je input ;if true then again taking the input

;comparing either the given ans either correct or not

cmp bh,al

je increament ;if true then it will increament the flag and loop length

inc cl ;else it will only increase the loop length

cmp cl,52 ;checking either the length is end or not

jg print ;if true then jmp to print how many are right

jmp input ;jumping to the input label to take next input

increament:

;label for increamenting the flag and loop length

inc bl

inc cl

cmp cl,52 ;checking either the length is end or not

jg print ;if true then jmp to print how many are right

jmp input ;else again taking input

print:

;Student correct ans number means flag

lea dx,how\_correct

mov ah,9

int 21h

;prtinting the flag value bl

mov ah,2

mov dl,bl

int 21h

jmp program\_start ;again starting the programme to take input

exit:

;exiting the programme

mov ah,4ch

int 21h

main endp

end main

**UVA 13012 C**

#include<stdio.h>

int main()

{

int sum,i,a,b;

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

{

sum=0;

for(i=0;i<5;i++)

{

scanf("%d",&b);

if(a==b)

sum++;

}

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

}

return 0;

}

**UVA 13018 Assembly**

include "emu8086.inc"

.model small

.stack 100h

.data

m db ?

n db ?

i db ?

t db ?

.code

main proc

;fetching all data

mov ax,@data

mov ds,ax

mov t,1 ;t=1;

again:

xor dl,dl

call input ;scanf("%d",&n)

mov m,dl

printn

xor dl,dl

call input ;scanf("%d",&n)

mov n,dl

mov bl,t ;if(t==0)

cmp bl,0

je one\_line\_print ;then print one line

jmp next\_step

one\_line\_print:

printn ;printf("\n");

next\_step:

mov bl,m

cmp bl,n

jg just\_exchange ;if(m>n)

jmp just\_exchange\_skip ;else skip exchanging

just\_exchange:

mov bl,m

mov bh,n

mov m,bh ;m=n

mov n,bl ;n=m

just\_exchange\_skip:

mov bl,m

cmp bl,n ;if(m==n)

je print\_m

jmp looping\_print\_n

print\_m:

add bl,1

cmp bl,9

jg greater1

mov dl,bl

add dl,48 ;printf("%d\n",m+1); for one digit

mov ah,2

int 21h

jmp after\_print\_m

greater1:

mov al,bl

mov ah,0

mov bl,10

div bl

mov cx,ax

mov dl,cl ;;printf("%d\n",m+1); for two digit

add dl,48

mov ah,2

int 21h

mov dl,ch

add dl,48

mov ah,2

int 21h

after\_print\_m:

printn ;print new line

jmp making\_t\_zero

looping\_print\_n:

mov bl,m

add bl,1

mov i,bl ;for(i=m+1;)

for\_loop:

mov bh,n

add bh,1

cmp i,bh ;i<=n+1

jg making\_t\_zero

cmp bl,9

jg greater2

mov dl,i

add dl,48 ;printf("%d\n",i); for one digit

mov ah,2

int 21h

jmp loop\_again

greater2:

mov al,i

mov ah,0

mov bl,10

div bl

mov cx,ax

mov dl,cl

add dl,48 ;;printf("%d\n",i); for 2 digit

mov ah,2

int 21h

mov dl,ch

add dl,48

mov ah,2

int 21h

loop\_again:

printn

inc i ;;i++

jmp for\_loop

making\_t\_zero:

mov t,0 ;t=0;

jmp again

main endp

input proc

input\_loop:

mov ah,1

int 21h

cmp al,13 ;taking input and checking either new line

je return

sub al,48

mov bl,al

mov al,dl

mov bh,10 ;taking multi digit by adding with 10\*before

mul bh ;+current

add al,bl

mov dl,al

jmp input\_loop

return:

ret

input endp

end main

**UVA 13018 C**

#include<stdio.h>

int main()

{

int m,n,temp,i,t=1;

while(scanf("%d%d",&m,&n)==2)

{

if(t==0)

printf("\n");

if(m>n){

temp=m;

m=n;

n=temp;

}

if(m==n)

printf("%d\n",m+1);

else

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

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

t=0;

}

return 0;

}

**UVA 13025 Assembly**

org 100h

.model small

.stack 100h

.data

date db "May 29, 2013 Wednesday",10,13,"$"

;defining the desired date

.code

main proc

;fetching the data

mov ax,@data

mov ds,ax

;printing the result

lea dx,date

mov ah,9

int 21h

;exiting the program

mov ah,4ch

int 21h

main endp

end main

**UVA 13025 C**

#include<stdio.h>

int main()

{

printf("May 29, 2013 Wednesday\n");

return 0;

}