# 31: UVA\_458\_Assembly

```
include 'emu8086.inc'
.model small
.stack 100h
.data
  char db 1000 DUP(?)
.code
main proc
 mov ax, @data
 mov ds, ax
 mov bx, 0; bx=0
Input:
          ;get the string
 mov ah, 1; input
 int 21h
 cmp al, 13
 je after ;if enter then exit
 sub al, 7
 mov char[bx], al
 inc bx
           ;bx++
 jmp Input ;jumpt to input
after:
  mov si, 0; si=0
  printn ;newline
Output:
            ;output
  cmp si, bx ;si=bx then exit
  je exit
  mov dl, char[si]
  mov ah, 2
  int 21h
  inc si
           ;si++
  jmp output
exit:
  mov ah, 4ch
  int 21h
  main endp
```

end main

# 31: UVA\_458\_C

```
#include<stdio.h>
int main()
{
    char ch[10000];
    int length, i;
    while(gets(ch))
    {
        length=strlen(ch);
        for(i=0; i<length; i++)
        {
            ch[i]=ch[i]-7;
        }
        puts(ch);
    }
    return 0;
}</pre>
```

#### cmp cx, 1 32. UVA\_494\_Asm ine after inc count ;count++ include 'emu8086.inc' after: .model small mov word, 0 ;word=0 .stack 100h jmp for\_last ;jump to last .data ; data section char db 100 dup(?) check az: $\inf(ch[i] > = 'a' \&\& ch[i] < = 'z')$ length dw? mov al, char[bx] word dw? cmp al, 97 count dw? il else i dw? cmp al, 122 char1 db? jg else .code jmp if ; jump to if main proc :word=0 else: mov ax, @data mov word, 1 mov ds, ax for last: while: inc bx ;bx++ mov bx, 0 inc i ;i++ Input\_string: ; input string jmp for mov ah. 1 exit for: int 21h mov ax, count ;ax=count mov char1, al call print\_num ;output count cmp al, 13 ;if press enter then exit from printn input jmp while ;jump to while je exit\_input\_string mov al, char1 exit: mov char[bx], al; insert into array mov ah, 4ch inc bx int 21h jmp Input\_string main endp exit\_input\_string: define\_print\_num ; define build in function printn define\_scan\_num mov length, bx; length=bx define\_print\_num\_uns mov count, 0 ;count=0 end main mov word, 1 ;word=1 mov i, 0 ;i=0mov bx, 0 bx=0for: mov ax, i cmp ax, length ;if i = length hen exitje exit\_for $\inf((ch[i] > = 'A' \&\& ch[i] < = 'Z')$ mov al, char[bx] cmp al, 65 il check az cmp al, 90 jg check\_az if: ;if(word)

mov cx, word

# 32. UVA\_494\_C

```
#include<stdio.h>
int main()
  char ch[10000];
  int i, count, word;
  while(gets(ch))
     count=0;
     word=1;
     for(i=0;\,ch[i]!='\backslash 0';\,i++)
        if((ch[i] >= 'A' \ \& \& \ ch[i] <= 'Z') \ \| \ (ch[i] >= 'a'
&& ch[i]<='z'))
           if(word){
           count++;
           }
           word=0;
        }
        else
           word=1;
        }
     }
     printf("%d\n", count);
  return 0;
```

# 33. UVA\_913\_Asm

```
include 'emu8086.inc'
.model small
.stack 100h
.data
  n dw?
  sum dw?
  odd dw?
.code
main proc
  mov ax, @data
  mov ds, ax
while:
  call scan_num ; input the number
  printn
  mov ax, cx
                ;ax=cx
  mov n, cx
                ;n=cx
  add ax, 2
               ;n+2
  mov cx, n
  mul cx
              ;n*(n+2)
  mov cx, 2
               ;cx=2
  div cx
             :ax/cx
  mov odd, ax
               ;odd=ax
  mov cx, 3
               ;cx=3
  mul cx
              ;ax=ax*cx
  sub ax, 6
              ;ax-ax
  call print_num ;print the number
  printn
jmp while
  main endp
           ; define buid in function
define_scan_num
define_print_num
define_print_num_uns
end main
```

# 33. UVA\_913\_C

```
#include <stdio.h>
int main()
{
    long int n,sum,odd;
    while(scanf("%ld",&n)==1)
    {
      odd=(n*(n+2))/2;
      sum=(3*odd)-6;
      printf("%ld\n",sum);
    }
return 0;
}
```

# 34.UVA\_1124\_ASM

```
include 'emu8086.inc'
.model small
.data
  str db 100 DUP (?)
  char db?
.code
main proc
while:
 mov bx, 0
for:
  mov ah, 1 ;input from user
  int 21h
  mov char, al ;char=al
  cmp al, 13
  je exit_for ;if enter then exit
  mov al, char
  mov str[bx], al ;str[bx]=char
  inc bx
             ;bx++
  jmp for
             ;jump to for
exit_for:
 printn
             ;newline
 mov si, bx
              ; si=bx
 mov bx, 0
               ;bx=0
for2:
  mov cx, bx
  cmp cx, si ;compare cx, si
  je after
            ; jump equal
  mov ah, 2
  mov dl, str[bx]; output
  int 21h
  inc bx
  jmp for2
               ; jump to for2
after:
  printn
  jmp while
               ; jump to while
Exit:
  mov ah, 4ch
  int 21h
  main endp
end main
```

# 34.UVA\_1124\_C

```
#include <stdio.h>
int main()
{
    char a[100];
    while(gets(a))
    puts(a);
    return 0;
}
```

35.UVA_10082_ASM	cmp al, 'Y' je pT
include 'emu8086.inc'; include library function .model small .stack 100h .data; data section	mov al, s[bx] cmp al, 'U' je pY
char db 1000 dup(?) length dw ? i dw ?  .code ; code section	mov al, s[bx] cmp al, 'I' je pU
main proc ; main proc start mov ax, @data; import data mov ds, ax	mov al, s[bx] cmp al, 'O' je pI
while: mov bx, 0	mov al, s[bx] cmp al, 'P' je pO
string_input: ; input string mov ah, 1 int 21h cmp al, 13 je exit_s_i	mov al, s[bx] cmp al, 'S' je pA
char[bx], al inc bx jmp string_input exit_s_i:	mov al, s[bx] cmp al, 'D' je pS
mov length, bx ;length=bx mov i, 0 ;i=0 for: mov ax, i ;ax=i amp ax length tempera ax length	mov al, s[bx] cmp al, 'F' je pD
cmp ax, length ;compare ax, length je exit_for ;switch, case  mov al, s[bx]	mov al, s[bx] cmp al, 'G' je pF
cmp al, 'W' je pQ  mov al, s[bx]	mov al, s[bx] cmp al, 'H' je pG
cmp al, 'E' je pW mov al, s[bx]	mov al, s[bx] cmp al, 'J' je pH
cmp al, 'R' je pE mov al, s[bx]	mov al, s[bx] cmp al, 'K' je pJ
cmp al, 'T' je pR  mov al, s[bx]	mov al, s[bx] cmp al, 'L'

je pK	je p6
mov al, s[bx] cmp al, 'X' je pZ	mov al, s[bx] cmp al, 8 je p7
mov al, s[bx] cmp al, 'C' je pX	mov al, s[bx] cmp al, 9 je p8
mov al, s[bx] cmp al, 'V' je pC	mov al, s[bx] cmp al, 0 je p9
mov al, s[bx] cmp al, 'B' je pV	mov al, s[bx] cmp al, '-' je p0
mov al, s[bx] cmp al, 'N' je pB	mov al, s[bx] cmp al, '=' je pHypen
mov al, s[bx] cmp al, 'M' je pN	mov al, s[bx] cmp al, '[' je pP
mov al, s[bx] cmp al, '2' je p1	
mov al, s[bx] cmp al, '3' je p2	mov al, s[bx] cmp al, ']' je pSthird
mov al, s[bx] cmp al, 4	mov al, s[bx] cmp al, '\' je pFthird
je p3  mov al, s[bx]  emp al, 5	mov al, s[bx] cmp al, ';' je pL
cmp al, 5 je p4	mov al, s[bx] cmp al, 1 je pBef1
mov al, s[bx] cmp al, 6 je p5	mov al, s[bx] cmp al, ',' je pM
mov al, s[bx] cmp al, 7	mov al, s[bx]

cmn at	
cmp al, '.' je pComma	pK:
je peomina	print "K"
mov al, s[bx]	jmp after
cmp al, '/'	jiip arter
je pDot	pH:
је рвог	print "H"
mov al, s[bx]	_
cmp al, ''	jmp after
	pZ:
je pSpace	•
neint "."	print "Z"
print ";"	jmp after
jmp after	n.I.
"C.	pJ:
pS:	print "J"
print "S"	jmp after
jmp after	VI.
pD:	pV:
print "D"	print "V"
jmp after	jmp after
pF:	
print "F"	
jmp after	pX:
pG:	print "X"
print "G"	jmp after
jmp after	a
nA:	pC: print "C"
pA:	
nrint "A"	
print "A"	jmp after
jmp after	jmp after
jmp after pO:	jmp after pB:
jmp after pO: print "O"	jmp after pB: print "B"
jmp after pO: print "O" jmp after	jmp after pB:
jmp after pO: print "O" jmp after pI:	jmp after pB: print "B" jmp after
jmp after pO: print "O" jmp after pI: print "I"	jmp after pB:     print "B"     jmp after pN:
jmp after pO: print "O" jmp after pI: print "I" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"
jmp after pO: print "O" jmp after pI: print "I" jmp after pU:	jmp after pB:     print "B"     jmp after pN:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U"	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY:	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y"	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT:	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T"	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after pR:	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after pR: print "R"	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"     jmp after
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after pR: print "R" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"     jmp after  p3:
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after pR: print "R" jmp after pE:	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"     jmp after  p3:     print "3"
jmp after pO: print "O" jmp after pI: print "I" jmp after pU: print "U" jmp after pY: print "Y" jmp after pT: print "T" jmp after pR: print "R" jmp after	jmp after  pB:     print "B"     jmp after  pN:     print "N"     jmp after  p1:     print "1"     jmp after  p2:     print "2"     jmp after  p3:

p4: print "4" jmp after	pL:     print "L"     jmp after
p5: print "5" jmp after	pBef1: print "`" jmp after
p6: print "6"	pM: print "M" jmp after
jmp after p7: print "7"	pComma: print "," jmp after
jmp after	pdot: print "." jmp after
p8: print "8" jmp after	pSpace:     print " "     jmp after
p9: print "9" jmp after	pQ: print "Q" jmp after
p0: print "0" jmp after	after:     printn     int i jmp for
pHypen: print "-" jmp after	exit_for: jmp while
pP:     print "P"     jmp after pSthird:     print "["     jmp after	exit: mov ah, 4ch ; exit int 21h main endp ; main proc end define_print_num_uns define_print_num define_scan_num end main
pFthird:     print "]"     jmp after pW:     print "W"     jmp after	

35.UVA_10082_C	<pre>printf("V"); case 'N':</pre>	break;
#include <stdio.h></stdio.h>	<pre>printf("B");</pre>	break;
#include <string.h></string.h>	case 'M':	
int main(){	<pre>printf("N");</pre>	break;
long int length,i; char s[100000];	case '2':	
while(gets(s)){	<pre>printf("1");</pre>	break;
length=strlen(s);	case '3':	
$for(i=0;i < length;i++){$	<pre>printf("2");</pre>	break;
switch(s[i]){	case '4':	
case 'W':	<pre>printf("3");</pre>	break;
<pre>printf("Q"); break;</pre>	case '5':	
case 'E':	<pre>printf("4");</pre>	break;
<pre>printf("W"); break;</pre>	case '6':	
case 'R':	<pre>printf("5");</pre>	break;
printf("E"); break;	case '7':	
case 'T':	<pre>printf("6");</pre>	break;
printf("R"); break;	case '8':	,
case 'Y':	printf("7");	break;
printf("T"); break;	case '9':	,
case 'U':	printf("8");	break;
printf("Y"); break;	case '0':	or <b>cu</b> n,
case 'I':	printf("9");	break;
printf("U"); break;	case '-':	orean,
case 'O':	printf("0");	break;
printf("I"); break;	case '=':	oreak,
case 'P':	printf("-");	break;
	case '[':	oreak,
printf("O"); break;	printf("P");	break;
case 'S':	case ']':	oreak,
printf("A"); break;	printf("[");	break;
case 'D':	case '\\':	oreak,
printf("S"); break;	printf("]");	hraals
case 'F':	_	break;
printf("D"); break;	case ';':	1
case 'G':	<pre>printf("L");</pre>	break;
printf("F"); break;	case '1':	1 1
case 'H':	<pre>printf("`");</pre>	break;
printf("G"); break;	case ',':	1 1
case 'J':	<pre>printf("M");</pre>	break;
printf("H"); break;	case '.':	1 1
case 'K':	<pre>printf(",");</pre>	break;
printf("J"); break;	case '/':	1 1
case 'L':	<pre>printf(".");</pre>	break;
<pre>printf("K"); break;</pre>	case ' ':	
case 'X':	<pre>printf(" ");</pre>	break;
<pre>printf("Z"); break;</pre>	default:	
case 'C':	<pre>printf(";");</pre>	break;
<pre>printf("X"); break;</pre>	} }	
case 'V':	<pre>printf("\n"); }</pre>	
<pre>printf("C"); break;</pre>	return 0;	
case 'B':	}	

# 36.UVA\_10469\_Asm

```
include 'emu8086.inc'
.model small
.stack 100h
.data
               ; data section
  a dw?
.code
main proc
while:
  call scan_num
  mov a, cx
                  ;cin>>a
  printn
  call scan_num :cin>>b
  printn
  xor cx, a
                ;a xor b
  mov ax, cx
  call print_num
  printn
  jmp while
  mov ah,4ch
  int 21h
  main endp
           ; define build in function
define_scan_num
define_print_num
define_print_num_uns
end main
```

# 36.UVA\_10469\_C

```
#include <bits/stdc++.h>
using namespace std;

int main()
{
   int a,b;
   while(cin >> a >> b)
   {
      cout << (int)(a^b) << endl;
   }
   return 0;
}</pre>
```

# 37.UVA\_10696\_Asm

```
include 'emu8086.inc'; include library function
.model small
.stack 100h
.data
            ; data section
N 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, cx
    printn
     cmp \ cx, 0; compare \ cx \ and \ 0
     je exit
     cmp cx, 100
             ; if greater than 100 then jump to if
     print "f91("
     mov ax, N
     call print_num
     printn ") = 91"; printf("f91(%d) = 91\n", N);
     imp while
 if:
             printf("f91(%d) = %d\n", N, N-10);
     print "f91("
     mov ax, N
     call print_num
     print ") = "
     mov ax, N
     sub ax, 10
     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
```

### 37.UVA\_10696\_C

```
#include<stdio.h>
int main()
{
    int N;
    while(scanf("%d", &N)==1)
    {
        if(N==0)
            break;

        if(N>100)
            printf("f91(%d) = %d\n", N, N-10);
        else
            printf("f91(%d) = 91\n", N);
        }
        return 0;
}
```

38.UVA_11498_Asm	jmp after else_if1: ;else if (x < px && y > py)
include 'emu8086.inc'	mov ax, x
.model small	cmp ax, px
.data	jl check_py
n dw ?	jmp elseif2
px dw ?	•
py dw?	check_py:
x dw?	mov ax, y
y dw?	cmp ax, py
.code	jg print_NO
main proc	jmp elseif2
call scan_num ;cin>>n	3 1
mov n, cx	print_NO:
printn ;newline	printn "NO"
while:	jmp after
	July witte
mov cx, n ; compare n and 0	elseif2: ;else if $(x > px & y > py)$
cmp cx, 0	mov ax, x
je Exit	cmp ax, px
2211 2224 2224	jg check_py2
call scan_num	jmp elseif3
mov px, cx ; input px	Jinp ciscus
printn	check_py2:
	mov ax, y
call scan_num; input py	cmp ax, y
mov py, cx	jg print_NE
printn	jmp elseif3
1.11.0	print_NE:
while2:	print_NE"
mov cx, n	jmp after
$\operatorname{cmp} \operatorname{cx}, 0$	Jinp arter
je after2	elseif3: ;else if $(x > px && y < py)$
**	
call scan_num	mov ax, x
mov x, cx ; input x	cmp ax, px
printn	jg check_py3
	jmp elseif4
call scan_num	about my2.
mov y, cx ;input y	check_py3:
printn	mov ax, y
$; if (x == px \parallel y == py)$	cmp ax, py
mov ax, x	jl print_SE
cmp ax, px	jmp elseif4
je print_divisa	print_SE:
mov ax, y	printn "SE"
cmp ax, py	jmp after
je print_divisa	1-264.
jmp else_if1	elseif4: ;else if $(x < px && y < py)$
	mov ax, x
print_divisa:	cmp ax, px
printn "divisa"	jl check_py4

```
jmp after
                                                          38.UVA_11498_C
check_py4:
                                                          #include <iostream>
  mov ax, y
                                                          using namespace std;
  cmp ax, py
  jl print_SO
                                                          int main() {
  jmp after
                                                                  int n, px, py, x, y;
print_SO:
  printn "SE"
                                                                  cin >> n;
                                                                  while (n != 0) \{
after:
                                                                          cin >> px >> py;
   dec n
   printn
             ; newline
                                                                          while (n--) {
  jmp while2 ;jump to while
                                                                                  cin >> x >> y;
after2:
                                                                                  if (x == px || y == py)
  jmp while
                 ; jump to while
                                                                                          cout << "divisa";</pre>
                                                                                  else if (x < px & y > py)
 Exit:
                                                                                          cout << "NO";
  mov ah, 4ch
                                                                                  else if (x > px & y > py)
  int 21h
                                                                                          cout << "NE";
                                                                                  else if (x > px && y < py)
  main endp
                                                                                          cout << "SE";
define_scan_num
                                                                                  else if (x < px && y < py)
define_print_num
                                                                                          cout << "SO";
define_print_num_uns
                                                                                  cout << endl;</pre>
end main
                                                                          }
                                                                          cin >> n;
                                                                  return 0;
                                                          }
```

39.UVA_11526_Asm	Exit:
	mov ah, 4ch
include 'emu8086.inc'	int 21h
.model small	
.stack 100h	main endp
.data	;define build in function
T dw?	define_scan_num
i dw?	define_print_num
n dw?	define_print_num_uns
res dw?	end main
x dw?	
.code	
main proc	39.UVA_11526_C
mov ax, @data	
mov ds, ax	#include <stdio.h></stdio.h>
call scan_num; input T	long long H(int n)
mov T, cx	{
printn ;newline	long long res $= 0$ ;
while:	int i;
mov ax, T	$for(i = 1; i \le n; i++)$
cmp ax, 0 ;compare	{
je exit ; $T==0$ then exit	res = (res + n/i);
call scan_num; input x	}
mov x, cx	return res;
printn ;newline	}
mov res, 0 ;res=0	int main()
mov i, 1 ; $i=1$	{
for: ;for( $i = 1$ ; $i \le n$ ; $i++$ )	int $T, x, j$ ;
mov ax, i	long long result;
cmp ax, x	scanf("%d", &T);
jg exit_for	while(T)
	{
xor dx, dx	scanf("%d", &x);
mov ax, x	result=H(x);
mov cx, i	<pre>printf("%lld\n", result);</pre>
div cx ;n/i	}
add ax, res ; $ax=(res + n/i)$	return 0;
mov res, ax ; res = $ax$	}
inc i ;i++	
jmp for	
exit_for: ;displying result	
mov ax, res	
call print_num	
printn	
printin	
dec T ; T	
jmp while ;jump to while	

40.UVA_11839_Asm	mov ax, count ;ax=count
: . 1. 1. 1	cmp ax, 1 ;if(count==1) jne else
include 'emu8086.inc' .model small	jiie eise
stack 100h	mov ax, k
.data	cmp ax, 1 ; $f(k==1)$
N dw ?	je printA
A dw 6 DUP(?)	cmp ax, 2 ; $f(k==)$
i dw?	je printB
k dw?	cmp ax, 3 ; $f(k==3)$
j dw ?	je printC
count dw?	cmp ax, 4 ; $f(k==4)$
.code	je printD
main proc	cmp ax, 5 ; $f(k==4)$
mov ax, @data; data import	je printE
mov ds, ax	jmp after2
1110 / 655 621	printA:
while:	printn "A" ;display A
call scan_num ;input N	jmp after2
mov N, cx	printB:
printn	printn "B" ;display B
cmp cx, 0	jmp after2
ile while	printC:
·	printn "C" ;display C
mov i, 0 ; i=0	jmp after2
	printD:
for1:	printn "D" ;display D
mov ax, i	jmp after2
cmp ax, N ;compare i and N	printE:
jge exit_for1	printn "E" ;display E
mov count, 0 ;count=0	jmp after2
mov j, 1	
for2:	
mov ax, j ; ax=j	else:
cmp ax, 5; $ax=5$	printn "*" ;display *
jg exit_for2	
call scan_num; input a number	after2:
printn	inc i ;i++
mov bx, j	jmp for1
mov A[bx], $cx$ ; insert into A[j]	exit_for1:
cmp cx, 127	jmp while ;jump to while
jg after	P :
inc count ;count++	Exit:
mov ax, j ; ax=j	mov ah, 4ch
mov k, ax ;k=ax	int 21h
after:	main endp
inc j	; define build in function
jmp for2	define_scan_num
	define_print_num
exit_for2:	define_print_num_uns end main
	ena mam

# 40.UVA\_11839\_C

```
#include<stdio.h>
int main()
  int N, A[6], i, k, j;
  while(scanf("%d", &N) == 1)
     if(N>0){
     for(i=0; i<N; i++)
       int count=0;
       for(j=1; j<=5; j++)
         scanf("%d", &A[j]);
          if(A[j] \le 127)
            count++;
            k=j;
          }
       if(count==1){
       if(k==1)
          printf("A\n");
       else if(k==2)
          printf("B\n");
       else if(k==3)
          printf("C\n");
       else if(k==4)
          printf("D\n");
       else if(k==5)
          printf("E\n");
       }
       else
          printf("*\n");
     }
  return 0;
```

include 'emu8086.inc'; include library function .model small .stack 100h .data ; data section    T dw?    i dw?    md w?    k dw?    Ad b DUP 100 (0)    num dw? .code ; code section    main proc ; main proc start    mov ax, @data; import data    mov dx, ax  while:    call scan_num ; input T    mov T, cx    printn    mov m, 1    for: ; for start    mov ax, m ; ax=m    cmp ax, T ; compare m and T jig exit for    call scan_num; input i    mov bx, 0 ; k=0    mov bx, 0 ; k=0    mov bx, 0; k=0    int Ti, m, k;    int A[100]; while(scanf("%d", &T)==1)    {    for(k=0; k <i; %d:="" %d\n",="" &a[k]);="" 2])="" 2])<="" a[k="" k++)="" m,="" printf("case="" scanf("%d",="" th="" {="" }=""><th>41.UVA_11875_Asm</th><th>mov ax, k</th></i;>	41.UVA_11875_Asm	mov ax, k
model small stack 100h		mov cx, 2
.model small	include 'emu8086.inc' : include library function	div cx ; k/2
.stack 100h .data ; data section  T dw? i dw? i dw? m dw? k dw? A db DUP 100 (0) num dw? .code ; code section main proc ; main proc start mov ax, @data; import data mov ax, @data; import data mov dx, ax while: call scan_num ; input T mov T, cx printn mov m, 1 for: ;for start mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num ;input i mov bx, 0 ;k=0 mov bx, 0 ;bx=0 for2: mov ax, k ;ax=k cmp ax, i ;compare k and i je exit_for2 call scan_num ;input A[k] printn ; newline mov bx, k ;bx=k mov A[bx], cl ;insert value into array inc k ;k++ jmp for2: printn ; dispaying result print ; dispaying result pr	•	mov bx. ax
.data (data section and the se		•
T dw ? i dw ? m dw ? k dw ? A db DUP 100 (0) num 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, ex printn mov m, 1 for: ; for start mov ax, m iax=m cmp ax, T ; compare m and T je skit_for call scan_num ; input i mov k, 0 ; k=0 mov bx, 0 ; bx=0 for 2: mov ax, k ; ax=k cmp ax, i ; compare k and i je exit_for2 call scan_num; input A[k] printn ; newline mov bx, k ; bx=k mov A[bx], cl ; insert value into array inc k ; k++ jmp for 2 jump to for 2 exit_for 2: printn ; dispaying result print "Case " mov ax, m call print_num printn  inc m ; m++ jmp for  imc m ; m++ jmp for  exit_for:  call print_num jmp for  exit_for:  exit tor:  call print_num jmp for  imc m ; m++ jmp while  exit: int 21h main endp ; main proc end define print_num define print_num define scan_num input A[t]  int main(){ int main(){ int main(){ int Ti, m, k; int A[100]; while(scanf("%d", &T)==1) { for(m=1; m<=T; m++) { scanf("%d", &A[k]); } } printf("Case %d: %d'\mathra m, A[k/2])		
i dw ?     m dw ?     kdw ?     A db DUP 100 (0)     num dw ?     code    ; code section     main proc    ; main proc start     mov ax,		
m dw ?         inc m ;m++           A db DUP 100 (0)         jmp for           num dw ?         exit_for:           .code ; code section         exit_for:           main proc ; main proc start         jmp while           mov ax, @data; import data         exit;           mov T, cx         define_print_num_uns           define_print_num         define_print_num           define_print_num         define_print_num           define_print_num         end main           and print_num         define_print_num           define_print_num         end main           and print_num         define_print_num           define_print_num         end main           and print_num         define_print_num           and print intent         #InCVA_11875_C           and print intent         #InCVA_11875_C           and print intent         #InCVA_11875_C           and print intent		-
k dw ?     A db DUP 100 (0)     num dw ?     exit_for:  .code    ; code section main proc    ; main proc start     mov ax,		printn
A db DUP 100 (0) num dw?  .code		
num 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 m, 1 for: ; for start mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num ; input i mov k, 0 ; k=0 mov bx, 0 ; kx=0 for2: mov ax, k ; ax=k cmp ax, i ; compare k and i je exit_for2 exit_for2 exit_for call scan_num ; input i mov bx, k ; bx=k mov ax, m call print_num; call print_num define_print_num define_print_num define_print_num define_scan_num int and p; main proc end define_print_num define_scan_num int ali print_num define_scan_num int ali print i main endp ; main proc end define_print_num d	k dw?	,
exit_for:  main proc ; main proc start mov ax, @data; import data mov ds, ax  while:  call scan_num ; input T mov T, cx printn mov m, 1 for: ; for start mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num ; input i mov h, 0 ; k=0 mov bx, 0 ; k=0 for2: mov ax, k ; ax=k cmp ax, i ; compare k and i je exit_for2 call scan_num ; input A[k] printn ; newline mov bx, k ; bx=k mov A[bx], cl ; insert value into array  inc k ; k++ jmp for2 ; jump to for2 exit_for2. exit_mov ah, 4ch ; exit int 21h main endp ; main proc end define_print_num define_scan_num end main  41.UVA_11875_C  #include <stdio.h> int main(){ int Ti, im, k; int A[100]; while(scanf("%d", &amp;T)==1) {     scanf("%d", &amp;T)==1)     {         scanf("%d", &amp;A[k]);         }         printf("Case %d: %d\n", m, A[k/2])     }     return 0; }  return 0;</stdio.h>	A db DUP 100 (0)	jmp for
.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 m, 1 for: ; for start mov ax, m rax=m cmp ax, T ; compare m and T jg exit_for call scan_num ; input i mov i, cx printn ; newline mov k, 0 ; k=0 mov bx, 0 ; bx=0 for 2: call scan_num ; input A[k] printn ; newline mov bx, k ; bx=k mov A[bx], cl ; insert value into array  call print_num; call print_num proc  mov ax, m call print_num; call print_num proc  exit: mov ah, 4ch ; exit int 21h main endp ; main proc end define_print_num define_scan_num define_print_num define_pri	num dw?	
main proc ; main proc start mov ax, @data; import data mov ds, ax exit:  while:  call scan_num ; input T mov T, cx printn mov m, 1  for: ; for start end main  for: all scan_num ; input i mov i, cx printn ; newline mov k, 0 ; k=0 mov bx, 0		exit_for:
main proc ; main proc start mov ax, @data; import data mov ds, ax exit:  while:  call scan_num ; input T mov T, cx printn mov m, 1  for: ; for start end main  for: all scan_num ; input i mov i, cx printn ; newline mov k, 0 ; k=0 mov bx, 0	.code : code section	
mov ax, @data; import data mov ds, ax  while: call scan_num; input T mov T, cx printn mov m, 1 for: ; for start mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num; input i mov h, 0; k=0 mov bx, 0; bx=0 for2: mov ax, k ; ax=k cmp ax, i ; compare k and i je exit_for2 call scan_num; input A[k] printn ; newline mov bx, k ; bx=k mov A[bx], cl; insert value into array inc k ; k++ jmp for2 ; jump to for2 exit_for2: exit_mov ax, m call print_num; call print_num proc  exit: mov ah, 4ch; exit int 21h main endp; main proc end define_print_num_uns define_print_num define_scan_num fend main  41.UVA_11875_C  41.UVA_11875_C  41.UVA_11875_C  41.UVA_11875_C  for(m=1; m<=1; m++) {		imp while
mov ds, ax  while: call scan_num ; input T mov T, cx printn mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num ; input i mov bx, 0 ; k=0 mov bx, 0 ; bx=0 for2: mov ax, k ; ax=k cmp ax, i ; compare k and i je exit_for2 call scan_num ; input A[k] printn ; newline mov bx, k ; bx=k mov A[bx], cl ; insert value into array  inc k ; k++ jmp for2 ; jump to for2 exit_for2: printn ; dispaying result print "Case" call print_num ; call print_num proc  exit: mov ah, 4ch ; exit int 21h main endp ; main proc end define_print_num define_scan_num define_scan_num define_scan_num define_scan_num define_scan_num end main  41.UVA_11875_C  41.UVA_1	•	J
while: call scan_num; input T mov T, cx printn mov ax, m; ax=m cmp ax, T; compare m and T jg exit_for call scan_num; input i mov k, 0; k=0 mov bx, 0; bx=0 for2: mov ax, k; ax=k cmp ax, i; compare k and i je exit_for2 call scan_num; input A[k] printn; newline mov bx, k; bx=k mov A[bx], cl; insert value into array inc k; k++ jmp for2; jump to for2 exit_for2: print; dispaying result print "Case" mov ax, m call print_num; input A[k] print_n; dispaying result print "Case" mov ax, m call print_num; input A[cl] print_num; input of rod define_print_num mov ah, 4ch; exit int 21h main endp; main proc end define_print_num define_scan_num define_print_num define_print_num define_print_num define_scan_num define_print_num d	-	evit:
while:     call scan_num ; input T     mov T, cx     printn     mov m, 1     for: ; for start     mov ax, m ; ax=m     cmp ax, T ; compare m and T     jg exit_for     call scan_num ; input i     mov k, 0 ; k=0     mov bx, 0 ; bx=0     for2:     mov ax, k ; ax=k     cmp ax, i ; compare k and i     je exit_for2     call scan_num ; input A[k]     printn ; newline     mov bx, k ; bx=k     mov A[bx], cl ; insert value into array     inc k ; k++     jmp for2 ; jump to for2     exit_for2:     printn ; dispaying result     print Case "     mov ax, m     call print_num; call print_num proc  int 21h     main endp ; main proc end     define_print_numuns     define_print_numuns     define_print_numuns     define_print_numuns     define_print_num     define_print_num     define_print_num     define_print_num     define_print_numuns     define_print_num     define_print_num     define_print_numuns     define_print_num     define_print_numuns     define_print_numuns     define_print_numuns     define_print_num     de	mov us, ax	
$ \begin{array}{cccccccccccccccccccccccccccccccccccc$	• • •	
mov T, cx         define_print_num_uns           printn         define_print_num_define_scan_num           for: ;for start         end main           mov ax, m         ;ax=m           cmp ax, T         ;compare m and T           jg exit_for         for luVA_11875_C           call scan_num ;input i         41.UVA_11875_C           mov i, cx         mov i, cx           printn ;newline         #include <stdio.h>           mov bx, 0         ;k=0         int main(){           mov bx, 0         ;k=0         int T,i, m, k;           for2:         int A[100];         while(scanf("%d", &amp;T)==1)           cmp ax, i         ;compare k and i         {           je exit_for2         for(m=1; m&lt;=T; m++)</stdio.h>		
$\begin{array}{llllllllllllllllllllllllllllllllllll$	* *	•
mov m, 1  for: ;for start  mov ax, m ;ax=m  cmp ax, T ;compare m and T  jg exit_for  call scan_num ;input i  mov k, 0 ;k=0  mov bx, 0 ;bx=0  for2:  mov ax, k ;ax=k  cmp ax, i ;compare k and i  je exit_for2  call scan_num ;input A[k]  printn ; newline  mov bx, k ;bx=k  mov bx, k ;bx=k  mov bx, k ;bx=k  mov A[bx], cl ;insert value into array  inc k ;k++  jmp for2 ;jump to for2  exit_for2:  printn ; dispaying result print "Case "  mov ax, m  call print_num ; call print_num proc  define_scan_num end main  41.UVA_11875_C  41.UVA_11875_C  41.UVA_11875_C  #include <stdio.h> int main(){ int T,i, m, k; int A[100]; while(scanf("%d", &amp;T)==1)  {</stdio.h>	mov T, cx	_
$\begin{array}{llllllllllllllllllllllllllllllllllll$	printn	define_print_num
mov ax, m ; ax=m cmp ax, T ; compare m and T jg exit_for call scan_num ; input i	mov m, 1	define_scan_num
mov ax, m ;ax=m cmp ax, T ;compare m and T jg exit_for call scan_num ;input i	for: ;for start	end main
$\begin{array}{llllllllllllllllllllllllllllllllllll$	•	
jg exit_for call scan_num ;input i mov i, cx printn ;newline mov k, 0 ;k=0 int main(){ mov bx, 0 ;bx=0 int A[100]; mov ax, k ;ax=k cmp ax, i ;compare k and i je exit_for2 call scan_num ;input A[k] printn ; newline  mov bx, k ;bx=k mov A[bx], cl ;insert value into array  inc k ;k++ jmp for2 ;jump to for2 exit_for2: printn ; dispaying result print "Case " mov ax, m call print_num ; call print_num proc  41.UVA_11875_C  #include <stdio.h> int main(){ int T,i, m, k; int A[100]; while(scanf("%d", &amp;T)==1)  { for(m=1; m&lt;=T; m++) { scanf("%d", &amp;i);  for(k=0; k<i; %d:="" %d\n",="" 0;<="" 2])="" a[k="" k++)="" m,="" printf("case="" return="" td="" }=""><td></td><td></td></i;></stdio.h>		
call scan_num ;input i     mov i, cx     printn ;newline		
mov i, cx printn ;newline  #include <stdio.h> mov k, 0 ;k=0  int main(){ mov bx, 0 ;bx=0  int A[100]; mov ax, k ;ax=k  while(scanf("%d", &amp;T)==1) cmp ax, i ;compare k and i</stdio.h>		41 UVA 11975 C
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	-	41.0 VA_110/3_C
mov k, 0		W 1 1 1 1 1
mov bx, 0 ;bx=0 int T,i, m, k; for2:	•	
$\begin{array}{llllllllllllllllllllllllllllllllllll$		· ·
$\begin{array}{llllllllllllllllllllllllllllllllllll$		int T,i, m, k;
$\begin{array}{llllllllllllllllllllllllllllllllllll$	for2:	
$\begin{array}{lll} \text{je exit\_for2} & \text{for}(\text{m=1; m<=T; m++}) \\ \text{call scan\_num ; input A[k]} & \{ \\ \text{printn} & ; \text{newline} & \text{scanf}(\text{"}\%\text{d", \&i}); \\ \\ \text{mov bx, k } & ; \text{bx=k} & \text{for}(\text{k=0; k$	mov ax, k ;ax=k	while( $scanf("\%d", \&T)==1)$
$\begin{array}{lll} \text{je exit\_for2} & \text{for}(\text{m=1; m<=T; m++}) \\ \text{call scan\_num ; input A[k]} & \{ \\ \text{printn} & ; \text{newline} & \text{scanf}(\text{"}\%\text{d", \&i}); \\ \\ \text{mov bx, k } & ; \text{bx=k} & \text{for}(\text{k=0; k$	cmp ax, i ;compare k and i	{
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	· · · · · · · · · · · · · · · · · · ·	for(m=1: m<=T: m++)
$\begin{array}{llllllllllllllllllllllllllllllllllll$	-	{
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	-	scanf("%d" &i):
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	princi , ne write	seum ( /// ( C1),
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	mov by k hv-k	$for(k-0.1 \times i.1 \times 1.1)$
$scanf("\%d", \&A[k]); \\ inc k ; k++ \\ jmp for 2 ; jump to for 2 \\ exit\_for 2: \\ printn ; dispaying result \\ print "Case" \\ mov ax, m \\ call print\_num ; call print\_num proc \\ \\ scanf("\%d", \&A[k]); \\ printf("Case %d: %d\n", m, A[k/2]) \\ printf("Case %d: %d\n", m, A[k/2]) \\ return 0; \\ return 0; \\ \\ call print\_num ; call print\_num proc \\ \\ \\ \\ call print\_num ; call print\_num proc \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\$		
$\begin{array}{cccccccccccccccccccccccccccccccccccc$	mov A[0x], ci , msert value mto array	•
<pre>jmp for2 ;jump to for2</pre>		scani (%d, &A[k]);
exit_for2:  printn ; dispaying result  print "Case " return 0;  mov ax, m  call print_num ; call print_num proc  }	•	}
printn ; dispaying result } print "Case " return 0; mov ax, m } call print_num ; call print_num proc		
print "Case " return 0; mov ax, m } call print_num ; call print_num proc		}
mov ax, m } call print_num; call print_num proc	printn ; dispaying result	}
mov ax, m } call print_num ; call print_num proc	print "Case "	return 0;
call print_num; call print_num proc	-	
		•
	· · · · · · · · · · · · · · · · · · ·	

# 42.UVA\_11934\_Asm

42.UVA_11934_Asm	1 14
	check4:
include 'emu8086.inc'	mov ax, d
.model small	cmp ax, 0; compare d and 0
.stack 100h	je Exit
.data ;data section	else:
a dw?	mov count, $0$ ; count= $0$
b dw?	mov i, 0   ;i=0
c dw?	for:
d dw?	mov ax, i
L dw?	cmp ax, L ;compare i and L
i dw ?	jg exit_for
re1 dw?	xor dx, dx ; dx=0
count dw?	mov ax, a
.code ;code section	mov cx, i
main proc	mul cx ;ax=a*i
mov ax, @data	mul cx ; $ax=a*i*i$
mov ds, ax	mov re1, ax ; re1=ax
while:	, ,
call scan_num	xor dx, dx ; dx=0
mov a, cx ;input a	mov ax, b
printn	mov cx, i
call scan_num	mul cx ;ax=b*i
mov b, cx ;input b	add ax, re1 ;ax=ax+re1
printn	add ax, c ; $ax+=c$
call scan_num	x $dx$ $dx$ $dx$ $dx$ $dx$ $dx$
	mov cx, d ;cx=d
mov c, cx ;input b\c	div cx ;ax/cx
printn	cmp dx, 0 ;compare reminder and 0
call scan_num	jne after
mov d, cx ;input d	inc count
printn	
call scan_num	after:
mov L, cx ;input L	inc i ;i++
printn	jmp for ;jump to for
	exit_for:
mov ax, a	
cmp ax, 0; compare a and 0	mov ax, count ;print value of count
je check2	call print_num
jmp else	printn
	jmp while ;jump to while
check2:	
mov ax, b	Exit:
cmp ax, 0	mov ah, 4ch
je check3; compare b and 0	int 21h
jmp else	main endp
	; define build in function
check3:	define_print_num
mov ax, c	define_print_num_uns
cmp ax, 0 ; compare c and 0	define_scan_num
je check4	end main
jmp else	
JK	

# 42.UVA\_11934\_C

```
#include<stdio.h>
int main()
  int a, b, c, d, L, i, f, count;
  while(1)
    scanf("%d%d%d%d%d", &a, &b, &c, &d,
&L);
    if(a==0 && b==0 && c==0 && d==0
&&L==0)
      break;
    else
      count=0;
      for(i=0; i<=L; i++)
         f = a*i*i+b*i+c;
         if(f\%d==0)
           count++;
      printf("%d\n", count);
  return 0;
```

43.UVA_12289_Asm	inc count
include 'emu8086.inc'; include library function	if4: mov ax, count
.model small	cmp ax, 2 ; if(count>=2)
.stack 100h	jl else4
.data ; data section	printn "2" ;print 2
T dw?	jmp after
str_s db 1000 dup (?)	else4: printn "1" ;cout<<1< <endl< td=""></endl<>
count dw 0	after: jmp while
length dw 0	exit:
.code ; code section	mov ah, 4ch ; exit
main proc ; main proc start	int 21h
mov ax, @data; import data	main endp ; main proc end
mov ds, ax	define_print_num_uns
call scan_num	define_print_num
mov T, cx	define_scan_num
printn	end main
while: mov ax, T ;input t	
cmp ax, 0	
je exit	43.UVA_12289_C
mov length, 0; length=0	
mov bx, 0	#include <bits stdc++.h=""></bits>
input_string:	
mov ah, 1 ;input string	using namespace std;
int 21h	int main()
cmp al, 13	{
je exit_input	int T;
mov str_s[bx], al; insert data into array	string s;
inc length ;length++	cin>>T;
inc bx ;bx++	while(T)
jmp input_string	{
exit_input:	cin>>s;
printn	int count=0;
mov ax, length	if(s.length() == 5)
cmp ax, 5 $if(s.length() == 5)$	cout<<3< <endl;< td=""></endl;<>
ine else	else
printn "3" ;cout<<3< <endl;< td=""><td>{</td></endl;<>	{
imp after	if(s[0]=='t')
else: mov bx, 0	count++;
mov al, str_s[bx]	if(s[1]=='w')
cmp al, 't' ; $if(s[0]=='t')$	count++;
$\lim_{n \to \infty} u_n \cdot v_n \cdot v_n \cdot v_n = v_n \cdot $	if(s[2]=='o')
inc count	count++;
if2: mov bx, 1	if(count>=2)
mov al, str_s[bx]	cout < 2 < endl;
cmp al, 'w' ;if(s[1]== 'w')	else
-	cout<<1< <endl;< td=""></endl;<>
jne if3 inc count	·
if3: mov bx, 2	}
mov al, str_s[bx] cmp al, 'o' ;if(s[2]=='o')	return 0;
-	}
jne if4	

# 44.UVA\_12403\_Asm

44.U VA_12403_ASM	C.
	after:
include 'emu8086.inc'	dec t ;t
.model small	jmp while
.stack 100h	
.data	
t dw?	exit:
ans dw?	mov ah, 4ch
char db 100 dup(?)	int 21h
num dw ?	main endp
	define build in function
.code	•
main proc	define_scan_num
mov ax, @data	define_print_num
mov ds, ax	define_print_num_uns
	end main
call scan_num	
mov t, cx ; input test case	
printn	44.UVA_12403_C
mov ans, 0 ;ans=0	
,,	#include <bits stdc++.h=""></bits>
while:	miletude (bits/stde   1.11)
mov ax, t	using namasnaga std.
	using namespace std;
cmp ax, 0 ; if $t=0$ then exit	
je exit	int main()
mov bx, 0 ; bx=0	{
input_string: ;sting input	int t;
mov ah, 1	cin >> t;
int 21h ;input letter	int ans $= 0$ ;
cmp al, 13	while(t)
je break_input; if newline then stop taking input	{
cmp al, ''	string s;
je break_input ;if space then stop taking input	cin >> s;
mov char[bx], al ;add in array	int num;
inc bx; bx++	•
	if(s[0] == 'd')
jmp input_string	{
	cin >> num;
break_input:	ans += num;
mov bx, 0   ;bx=0	}
cmp char[bx], 'd'; if char[0]=d	else
je if ;then jump to if	cout << ans << endl;
mov ax, ans ; else	}
printn ;newline	return 0;
call print_num ;print ans	}
printn ;newline	,
jmp after	
if: ;if	
print " "	
call scan_num ;taking new num	
add cx, ans ; add ans and num	
mov ans, cx ;ans+=num	
printn ;newline	

# 45.UVA\_12461\_Asm

```
INCLUDE 'emu8086.inc'; include library
function
.MODEL SMALL
.STACK 100H
.DATA
             ; data section
 num dw?
.CODE
             ; code section
MAIN PROC
                ; main proc start
  MOV AX, @DATA; import data
  MOV DS, AX
while:
  call scan_num ; input n
            ; print newline
  printn
  mov num, cx
              ; cmp n and 0
  cmp cx, 0
            ;if n=0 ten exit
  je Exit
  printn "1/2"
              ; print 1/2
jmp while
              ; jmp while
Exit:
  MOV AH, 4CH ; exit
  INT 21H
  MAIN ENDP ; main proc end
  DEFINE_PRINT_NUM_UNS ;define
  DEFINE_PRINT_NUM
  DEFINE_SCAN_NUM
END MAIN
```

### 45.UVA\_12461\_C

```
#include<stdio.h>
int main(){
          unsigned int n;
          while(scanf("%u", &n) == 1 && n!=0){
          printf("1/2\n");
          }
          return 0;
}
```

### 46.UVA\_12502\_Asm

```
include 'emu8086.inc'
.model small
.data
  t dw?
  a dw?
  ans1 dw?
  b dw?
  c dw?
.code
main proc
  call scan_num
  printn
  mov t, cx
while:
  mov ax, t
  cmp ax, 0
  je exit
  call scan_num; cin>a
  printn
  mov a, cx
  call scan_num ;cin>b
  printn
  mov b, cx
  call scan_num ;cin>c
  printn
  mov c, cx
         ;c*(2*a*b)/(a+b)
  mov ax, a
  mov cx, 2
  mul cx : cx=a*2
  sub ax, b ;cx=b
  mov cx, c
  mul cx
          ;ax=c*cx
  xor dx, dx; dx=0
  mov bx, a ;bx=a
  add bx, b; bx+=b
  div bx ;ax=ax/bx
        ;displaying result
  call print_num
  printn
  dec t
jmp while
exit:
  mov ah, 4ch
  int 21h
```

```
main endp
define_scan_num
define_print_num
define_print_num_uns
end main
```

### 46.UVA\_12502\_C

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    long long t,a,b,c;
    cin>>t;
    while(t--)
    {
        cin>>a>>b>>c;
        cout<<c*(2*a-b)/(a+b)<<endl;
    }
    return 0;
}</pre>
```

# 47.UVA\_12577\_Asm

```
include 'emu8086.inc'
.model small
.stack 100h
.data
 str db 100 dup(?)
 char db?
.code
main proc
while:
  mov bx, 0
input:
  mov ah, 1
               ; input char
  int 21h
  mov char, al ; char=al
  cmp al, 2Ah ; if * then exit
  je exit
  mov al, char ; char=newline then exit Input
  cmp al, 13
  je exit_input
  mov al, char
  mov str[bx], al; str[bx]=char
             ; bx++
  inc bx
  jmp input
exit_input:
  printn
  mov si, 0; si=0
  cmp str[si], 48H ;48H='H'
  je akbar
  printn "Hajj-e-Asghar"
  jmp after
akbar:
  printn "Hajj-e-Akbar"
after:
             ; jump while
jmp while
exit:
  mov ah, 4ch
  int 21h
  main endp
end main
```

# 47.UVA\_12577\_C

```
#include<bits/stdc++.h>
#define LL long long
using namespace std;
int main()
  string s;
  int i = 1;
  while(cin >> s && s != "*")
     cout << "Case "<< i<<": ";
     if(s[0] == 'H')
       cout << "Hajj-e-Akbar\n";</pre>
     else
       cout << "Hajj-e-Asghar\n";</pre>
     i++;
  }
  return 0;
}
```

# 48.UVA\_13012\_Asm

```
INCLUDE 'emu8086.inc'; include library
function
.MODEL SMALL
.STACK 100H
.DATA
              ; data section
 c dw?
 i dw?
 n dw?
 t dw?
             ; code section
.CODE
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
 imp 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
```

# 48.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;
}
```

	mov i, 0 ; i=0
49.UVA_13025_Asm	
	for:
INCLUDE 'emu8086.inc'; include library function	mov ax, i ;ax=i
MODEL SMALL	cmp ax, T ;ax=T
STACK 100H	je exit_for
DATA ; data section	-
CODE ; code section	call scan_num ;input s
MAIN PROC ; main proc start	mov s, cx ;s=cx
; diplaying result	printn
printn "May 29, 2013 Wednesday"	call scan_num ;input d
Exit:	mov d, cx
MOV AH, 4CH ; exit	printn ;newline
INT 21H	r
MAIN ENDP ; main proc end	mov ax, s ;ax=d
END MAIN	cmp ax, d ;compare s and d, s <d< td=""></d<>
	jg print_impossible
	J8 print_impossion
49.UVA_13025_C	xor dx, dx ; dx=0
	mov ax, s ;(s-d)%2!=0
#include <bits stdc++.h=""></bits>	sub ax, d
using namespace std;	mov cx, 2
int main()	div cx
{	cmp dx, 0
cout<<"May 29, 2013 Wednesday"< <endl;< td=""><td>jne print_impossible</td></endl;<>	jne print_impossible
return 0;	jmp else
}	July 6186
	print_impossible:
	printn "impossible" ;displaying impossible
50.UVA_10812_Asm	jmp after
	else:
include 'emu8086.inc'; include library function	xor dx, dx
model small	mov ax, s ;ax=s
stack 100h	sub ax, d ;ax=s-d
data; data section	mov cx, 2 ; c=2
T dw?	div cx ;ax=ax/cx
s dw?	mov y, ax ;y=ax
d dw?	37
x dw?	mov ax, s ;ax=s
y dw?	add ax, d ;ax $=$ d
i dw?	mov cx, 2
	div cx
code ; code section	mov x, ax
main proc ; main proc start	,
mov ax, @data; import data	mov ax, x ; displaying x
mov ds, ax	call print_num
	print " ;displaying space
while:	mov ax, y
call scan_num ; input T	call print_num; displaying y
mov T, cx	printn
printn	after:

```
inc i ;i++ ;i++
jmp for ; jump to for

exit_for:
jmp while ;jump to while

exit:
mov ah, 4ch ; exit
int 21h
main endp ; main proc end
;define build in function
define_print_num_uns
define_print_num
define_scan_num
end main
```

# 50.UVA\_10812\_c

```
#include<stdio.h>
int main()
{
    int T, s, d, x, y, i;
    while(scanf("%d", &T)==1)
    {
        for(i=1; i<=T; i++)
        {
            scanf("%d%d", &s, &d);
            if(s<d || (s-d)%2!=0)
            {
                 printf("impossible\n");
            }
            else {
                 y = (s-d)/2;
                 x = (s+d)/2;
                 printf("%d %d\n", x, y);
            }
        }
        return 0;
}</pre>
```