important I/O related matters.

do you know about return values ?

a return value is the value of a function after its execution.

say,

```
int fn(int a, int b)
{
      printf("%d " , a+b);

      if(a+b > 0)
            return 1;

      return 0;
}
```

if you call this function it will print the sum of a and b. but after that it will also hold a value.

say, we call the function like,

```
fn(11, 22);
```

so it will just show the sum 33. but if we call it like,

```
int r = fn(11, 22);
```

it will print 33 and also the value of r will be 1.

if we call like,

```
r = fn(10, -100);
```

the value of r will be 0;

do you know, when a function returns a value using return keyword, the later part of the functions code is directly omitted ?

read Sordar's book and Subeen's pdf to learn details about them if you don't know yet.

in contest, how the program is judged ?

when testing your program in the pc, you are using console. but what the judge does ? its a robot. there is already a prepared input file in the judge pc. when you submit the program the robot enters the test input one after another and matches with the prepared correct output file.

if any mismatch is found it tell wrong answer – WA.
if all the outputs are matched – accepted – AC

if the judge can not check all the output within the time limit because your program is too slow and needs a better algorithm it tells – TLE – time limit exceed

if the program crashes during execution it tells – runtime error -RE

runtime error is such error that the can not tell about it when you build the program. they occur when you run the program. it may occur if you forget to use & sign inside scanf, or using wrong format specifier inside scanf/printf, using negative index or using higher index than the array size.

how does the robot say the input is finished ?

if you are getting input from a file then there must be an end of the files content. this end is marked by a special character called EOF. in console it is written by pressing ctrl+z (in linux ctrl+d)

if the program finds the EOF , it understands that the input is finished and so the program is terminated.

run a program that takes an input and print ctrl+z / ctrl+d then see it is terminated.

important matters related to EOF and Input functions

when you are using scanf do you know how does it work ? it returns a value.

say,

```
int a, b, c, v;

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

printf(“%d”, v);
```

what will be the value of v if all input value of a, b and c are valid ?

if you enter like 1, then 2 then 3, v will be 3.

if you enter like 1, then 2 then 'r' (which is invalid) then v will be 2.

if you enter r(invalid) then it will terminate at the first input and v will be 0.

in contest no invalid input is given by the judge. so all time scanf has to return the sum of total number of variables it is working with.

do experiment with different types of variables like char[ ], char, int etc. with scanf and test its value for invalid and valid input.

another thing is EOF.

say a program is reading input and it has reached the end of the file or in console ctrl+z / ctrl+d is pressed. then what will happen ?

like the program,

```
int a;
int v = scanf(“%d”, &a);
printf(“%d”, v);
```

run the program. it is saying that, input the value of a and assign the value of scanf function into v. then print the value of v.

if you press ctrl+z / ctrl+d and press enter then will see -1. because the value of EOF is -1.

you will be clear about the usefulness of this article when reading the I/O sytles pdf.

some important functions

gets()
puts()
getchar()
putchar()

read about them from Habib's book, Subeen's pdf book and C standard library pdf.

short notes about them

gets is used to input string in a easy way

puts is used to output string in a easy way but always prints an extra new line after the string. do experiment.

getchar is used to input a single character from input.

putchar is used to print a single character.

a special property of scanf()

did you notice, when we are working with scanf , inside the double quote we can use spaces or no spaces.

say,

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

it will input 2 integer.

but what about

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

there is a space inside the two %d. what does it mean ? it means it will input an integer. then before the second integer it will omit any space or new line.

wirte like,

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

there is another space after the second %d.
what does it mean ?
it means it will take an integer input, omit any space / new line before the second one. and after the second one it will also any space / new line. so if you input like,

11 (space) 22 (enter)
or like,
11 (enter) 22 (enter)

in both cases the last enter after 22 is omitted and the program will not end.

another example:

```
char a, b;
scanf("%c%c", &a, &b);
printf("%c%c.", a, b);
```

input like,

a (space) b (enter)

output will be a.
why ? because, a and space are taken as input and shown. you will see a space before the dot.

now input like,

ab

output will be ab.

now put an space inside the two %c.
like,
```
scanf("%c %c", &a, &b);
```

now input like

a (space) b (enter)

or like,

a (enter) b (enter)

output will be like, ab.
in both case because you know why.

a special problem with new line and different kinds of variable input

when we are working with only integer input or only character input this problem is not faced. but if we are working with different types of mixed input it may occur a problem.

say,

```
int digit, i;
char letter;

for(i = 1; i <= 2; i++)
{
    scanf("%c %d", &letter, &digit);

    printf(".%c.%d\n", letter, digit);
}
```

input like,

a (space) 4 (enter)

you will see .a.4

then again type

a (space) 4 (enter)

then you will see

.(blank line)
.4

now, where is the second lines input a ?

the problem is created by the red marked enter. first the character a is taken, then digit 4 is taken. after that, you are pressing enter and it is also a character called new line. it is saved in the buffer for later use. so when you are entering the second line, first the new line character from the saved memory is assigned into letter variable. then the integer value is fetched and assigned into digit variable. do experiment and try to understand if not clear yet or read again.

how to fix this problem ?

the thing we have to do – if we can remove the first pressed enter / new line anyway.

there is a function called getchar() which takes only one character from the input buffer. what we will do we shall take the input but will not use it.

write the code in this way,

```
int digit, i;
char letter;

for(i = 1; i <= 2; i++)
{
    scanf("%c%d", &letter, &digit);

    getchar();

    printf(".%c.%d\n", letter, digit);
}
```

now give the previous input

a (space) 4 (enter)

you will see .a.4

then again type

a (space) 4 (enter)

then you will see .a.4 again

what is happening here is that – in the first line a is taken then 4 is also taken as input. then the saved new line is taken with getchar() but not used or assigned to anything. then the second line comes with another a and 4.

during contest this knowledge will help you a lot.