Fundamentals

Translators are used to translate high-level language to machine level or binary language.

compiler :-

- compiler will translate entire program at a time
- Debugging was difficult.
- It is ~~easier~~ faster

Interpreter :-

-> It will translate line-by-line if there was any error in the current line, it doesn't move to the next line until we rectify that error.

-> Debugging was easy.

-> It was slow.

what is python ?

python is a open-source, high-level, general purpose dynamically typed interpreted language, features are

-> simple and easy to learn.

-> programs written in python language was nearer to english language.

-> general purpose programming language.

-> By using python we can build application for different domains.

-> **open source :-**

No need to purchase any license to use it source code is available freely.

-> Interpreted language.

-> To improve the performance of interpreter, python added JIT (just in time) compiler.

-> Dynamically typed language.

Based on data we assign to the variable, type of variable will decide at run-time.

-> platform independent, portable.

-> programs written in one-platform can execute in other platform without changing anything.

-> Architecture neutral - any architecture python behaviour will remains same.

-> It provides rich-set of libraries to build applications easily.

-> It provides garbage collector to deallocate allocated memory automatically.

Number System

Decimal to other Number system :-

To convert decimal to other number system, divide given number with target radix.

Decimal to octal :-

$(54)_{10} = (66)_8$

$(97)_{10} = (141)_8$

$(255)_{10} = (377)_8$

8 | 54
6 - 6

8 | 97
8 | 12 - 1
1 - 4

8 | 255
8 | 31 - 7
3 - 7

$(512)_{10} = (1000)_8$

8 | 512
8 | 64 - 0
8 | 8 - 0
1 - 0

$(2586)_{10} = (5032)_8$

8 | 2586
8 | 323 - 2
8 | 40 - 3
5 - 0

Decimal to HexaDecimal :-

$(54)_{10} = (36)_{16}$

16 | 54
3 - 6

$(232)_{10} = (68)_{16}$

16 | 232
14 - 8

$(956)_{10} = (3BC)_{16}$

16 | 956
16 | 59 - C (12)
3 - (11) B

$(2527)_{10} = (9DF)_{16}$

16 | 2527
16 | 157 - (15) F
9 - (13) D

$(5872)_{10} = (16F0)_{16}$

16 | 5872
16 | 367 - 0
16 | 22 - (15) F
1 - 6

Decimal to Binary :-

$(25)_{10} = (11001)_2$

2 | 25
2 | 12 - 1
2 | 6 - 0
2 | 3 - 0
1 - 1

$(98)_{10} = (1100010)_2$

2 | 98
2 | 49 - 0
2 | 24 - 1
2 | 12 - 0
2 | 6 - 0
2 | 3 - 0
1 - 1

$(295)_{10} = (100100111)_2$

2 | 295
2 | 147 - 1
2 | 73 - 1
2 | 36 - 1
2 | 18 - 0
2 | 9 - 0
2 | 4 - 1
2 | 2 - 0
1 - 0

2^{12} 2^{11} 2^{10} 2^9 2^8 2^7 2^6 2^5 2^4
4096 2048 1024 512 256 128 64 32 16
2^{13}
8192

2^0 2^1 2^2 2^3
1 2 4 8

$(9186)_{10} = (10001111110010)_2$

2 | 9186

$(578)_{10} = (1001000010)_2$

2 | 578

2	9186	
2	4593	-0
2	2296	-1
2	1148	-0
2	574	-0
2	287	-0
2	143	-1
2	71	-1
2	35	-1
2	17	-1
2	8	-1
2	4	-0
2	2	-0
	1	-0

2	578	
2	289	-0
2	144	-1
2	72	-0
2	36	-0
2	18	-0
2	9	-0
2	4	-1
2	2	-0
	1	-0

other number system to decimal number system :-

$(d_n \text{ ----- } d_1 d_0)_r = (\quad)_{10}$

$d_n * r^n + \text{ ----- } d_1 * r^1 + d_0 * r^0$

octal to decimal number system :-

$(27)_8 = (23)_{10}$

$2 * 8^1 + 7 * 8^0$

$= 16 + 7$

$= 23$

$(123)_8 = (83)_{10}$

$1 * 8^2 + 2 * 8^1 + 3 * 8^0$

$64 + 16 + 3$

$= 83$

$(512)_8 = (330)_{10}$

$5 * 8^2 + 1 * 8^1 + 2 * 8^0$

$320 + 8 + 2$

$= 330$

$(2572)_8 = (1402)_{10}$

$2 * 8^3 + 5 * 8^2 + 7 * 8^1 + 2 * 8^0$

$1024 + 320 + 56 + 2$

$= 1402$

$(632)_8 = (410)_{10}$

$6 * 8^2 + 3 * 8^1 + 2 * 8^0$

$384 + 24 + 2$

$= 410$

Hexadecimal to decimal number system :-

$(32)_{16} = (50)_{10}$

$3 * 16^1 + 2 * 16^0$

$48 + 2$

$= 50$

$(1A)_{16} = (26)_{10}$

$1 * 16^1 + 10 * 16^0$

$16 + 10$

$= 26$

$(CA)_{16} = (202)_{10}$

$12 * 16^1 + 10 * 16^0$

$192 + 10$

$= 202$

A = 10
B = 11
C = 12
D = 13
E = 14
F = 15

$(F2)_{16} = (242)_{10}$

$15 * 16^1 + 2 * 16^0$

$240 + 2$

$= 242$

Binary to Decimal number system :-

$(110011)_2 = (51)_{10}$

$1*2^5 + 1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

$32 + 16 + 0 + 0 + 2 + 1$

51

2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
256	128	64	32	16	8	4	2	1

$(111000)_2 = (56)_{10}$

$32 + 16 + 8 + 0 + 0 + 0$

$= 56$

$(100010111)_2 = (279)_{10}$

$256 + 0 + 0 + 0 + 16 + 0 + 4 + 2 + 1$

$= 279$

$(1010101010)_2 = (682)_{10}$

$512 + 0 + 128 + 0 + 32 + 0 + 8 + 0 + 2 + 0$

$= 682$

literals or constants :-

These values are fixed, we cannot able to modify.

1. Numerical literals :-

(i) Integer constants :- All decimal numbers are considered as integer constants

ex:- 252, 916, 410

(ii) Real or float constants :-

Numbers with decimal points

ex:- -15.67, 3.15, 4.96

- It supports scientific notations also

ex:- $5.4e3 = 5.4 * 10^3 = 5400.0$

(5.4) e (3) — exponent

Mantisa

$7.96e4 = 7.96 * 10^4 = 79600.0$

$2.67e^{-5} = 2.67 * 10^{-5} = 0.0000267$

(iii) complex constants :-

ex:- $a+bj$, $3+4j$, $3-7j$, $5.6+2.9j$, $3j$

2. Boolean constants :- True and false

3. string constants :-

Any data enclosed within the quotes will be considered as a string constant.

ex:- 'Hello', "abc", "xyz", '23'

Note :-

python doesn't support character data. even single character is a string.

4. special constant :- None is a special constant (no value for none).

Identifiers :-

All users defined names are identifiers (or) name given to variable or object or class or function will be considered as a identifiers.

Rules to define identiers :-

1. Identifier should start with either alphabet or underscore.
2. After the first letter subsequent letters may be either alphabet or underscore or digits.
3. spaces, keywords, special symbols were not allowed.
4. python is a case sensitive language, there was a difference between lower case, upper case Alphabets.

variable :-

Named memory locations are variables (or) its value ~~for~~ can vary, so it is variable.

-> In python, variables are immortable.

-> Syntax to define the variable : varname = value.

ex :- a = 5

b = 4.86

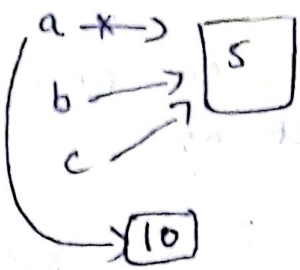
c = 'Hello'

a = 5

b = 5

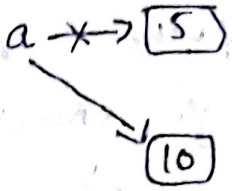
c = 5

a = 10



a = 5

a = 10



keywords :-

keywords are reserved words, these words functionality already defined in the translator, we need to use with some functionality.

printing a simple message :-

```
print ("Haneesha")
print ("253127")
```

operators :-

1. Arithmetic operators :-

operator	Meaning	example : a=8, b=5
+	add	a+b = 13
-	sub	a-b = 3
*	multiply	a*b = 40
/	float div	a/b = 1.6
//	floor(or) integer	a//b = 1
%	modular division	a%b = 3
**	power	a**b = 64

ex:- 19/5 = 3.8 | 27%.4 = 3 | 6 %. 8 = 6

19//5 = 3 | 27//4 = 6 | 6 // 8 = 0

19%.5 = 4 | 27/4 = 6.75 | 6 / 8 = 0.75

a = 25

b = 7

print ("sum of", a, ",", b, "is", a+b)

print (f"diff of {a}, {b} is {a-b}")

print ("prod of {}, {} is {}".format(a,b,a*b))

print ("float div of %.d, %.d is %.f" %.(a,b,a/b))

print (f'mod div of {a}, {b} is {a %.b}')

print (f"floor div of {a}, {b} is {a//b}")

print (f'{a} ** {b} = {a **b}')

Relational operators :- [<, >, <=, >=, ==, !=]

ex:-

5<12	12<=12	12!=15	15==12
True	True	True	False

logical operators :- [and, or, not]

To combine multiple ~~expressions~~ conditions as a single expression logical operators can use.

p	q	p and q	p or q	not p
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

write a expression to check a is smallest among a, b, c.

- a<b and a<c

write a expression to check n is divisible by either 3 or 5.

n%.3==0 or n%.5==0

All the languages will follow short circuit evaluation.

In the case of and operator if first expression result was false it doesn't evaluate second expression.

In the case of or operator if first expression result was true it will return the result of first expression itself. it doesn't evaluate second expression.

Note :- All non-zero values will consider as a true zero will consider as a false

ex: 3 and 5 — 5

3 and 0 — 0

0 and 5 — 0

5 and 8 and 12 — 12

ex: 3 or 5 — 3

0 or 6 — 6

Assignment operator :-

Assignment operator = L-value = R-value

L-value should be a single variable it should not be a constant or expression.

R-value may be a constant or expression or variable.

ex:- a = 5

b = a

c = a + b

Assigning same value to multiple variables

ex:- a = b = c = 20

a
20
b
20
c
20

or

Assigning different values to different variables.

ex:- a, b, c, d = 5, 8, 1, 2

a
5
b
8
c
1
d
2

write a program to swaping of 2 numbers

12

a = 5

b = 8

a, b = b, a

print (a, b)

a = 5.		a = 5
b = 8		b = 8
t = a	t = 5	a = a + b = 13
a = b	a = 8	b = a - b = 5
b = t	b = 5	a = a - b = 8

Short hand assignment :-

If L-value repeated immediate left in right side expression you can use short hand assignment.

ex:- $a = a + b$ $a = a + 1$ $a = a - 1$

$a += b$ $a += 1$ $a -= 1$

1. Write a python script to calculate area of a rectangle. $l \times b$
2. Calculate area of a circle. πr^2
3. Calculate perimeter of a circle. $2\pi r$
4. Calculate volume of a cube. s^3
5. Calculate area of a triangle. $\frac{1}{2}bh$
6. Calculate simple interest. $PxtxR/100$
7. extract last digit from a given number.
8. remove the last digit from a given number.
9. extract first two digits, last two digits from a given four digit number.
10. Calculate square root of a given number.

1.
```
l = int(input())
b = int(input())
print(l*b)
```

Output :- 3
2
= 6

2.
```
pi = 3.14
r = int(input())
print(pi*r*r)
```

O/P :-
= (3.14 * 2 * 2)
= 12.56

3. pi = 3.14
r = int (input())
print (2 * π * r)

o/p :- 2 * 3.14 * 3
= 18.84

4. s = int (input ("enter s value:"))
print (s**3)

o/p :-
enter s value : 3
= 27

5. r = int (input ("enter r value :"))
b = int (input ("enter b value :"))
h = int (input ("enter h value :"))
print (r * b * h)

o/p :- enter r value : 2
enter b value : 3
enter h value : 4
= 24

6. p = int (input())
t = int (input())
r = int (input())
print (p * t * r / 100)

o/p :- 2
3
4
= 0.24

7. n = int (input())
print (n % 10)

o/p :- 1234
4

8. n = int (input())
print (n // 10)

o/p :- 1234
123

9. n = int (input())
print (n // 100)
print (n % 100)

o/p :- 1234
12
34

10. n = int (input ())
print (n ** (1/2))

o/p :- 4

2.0

Unary operator :-
Unary - is used to represent negative sign (-)

ex :- a = -50

1. Note :- python doesn't support increment, decrement operators

Bitwise operators :-

and	&
or	|
xor	^
left shift	<<
right shift	>>
compliment	~

P	q	P & q	P | q	P ^ q
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

ex :-

	32	16	8	4	2	1	
a = 49	1	1	0	0	0	1	
b = 57	1	1	1	0	0	1	
a & b	1	1	0	0	0	1	-49
a | b	1	1	1	0	0	1	-57
a ^ b	0	0	1	0	0	0	-8

```
                64 32 16  8  4  2  1
a = 87 - -1  0  1  0  1  1  1   - 87
b = 18 -  0  0  1  0  0  1  0   - 18
-------------------------------------
a & b -  0  0  1  0  0  1  0    - 18
a | b -  1  0  1  0  1  1  1    - 87
a ^ b -  1  0  0  0  1  0  1    - 69
```

```
a = 87 - 1 0 1 0 1 1 1  - 87
b = 24 - 0 0 1 1 0 0 0
-----------------------
a & b - 0 0 1 0 0 0 0   - 16
a | b - 1 0 1 1 1 1 1   - 95
a ^ b - 1 0 0 1 1 1 1   - 79
```

left shift :-

If n left shift b (n << b) then result is $n \times 2^b$ or convert n into binary add b zeros at LSB side.

ex:- n = 12
b = 3

$n << b = n \times 2^b = 12 \times 2^3$
$= 96$

MSB ↖ LSB

n=12 → 1 1 0 0

b zeros

1 1 0 0 [0 0 0]

↳ 96

Right shift :-

If n >> b then result is $n // 2^b$ or convert n into binary format discard b bits from LSB side.

ex:- n = 20
b = 3

$n >> b = n // 2^b$
$= 20 // 3$
$= 2$

n=20 → 1 0 [1 0 0] discard b bits

10 → 2

Compliment :-

$\sim n = -(n+1)$

ex: n = 18

$\sim n = -(n+1)$

$= -(18+1)$

$= -19$

n = 18 — 0 0 0 1 0 0 1 0

1 1 1 0 1 1 0 1

n = 19 — 0 0 0 1 0 0 1 1

1 1 1 0 1 1 0 1 -> -19

Membership operators :- [in, not in]

To verify given value available within a sequence or not membership operator is used

ex: l = [4, 76, 12, 6, 15, 8, 7]

6 in l

True

13 in l

False

76 not in l

False

52 not in l

True

Identity operators :- [is, is not]

To verify two references or pointing to same objects or not this operators are used.

ex: a = 5

b = 5

a is b

True

```
l = [1, 2, 3, 4]
m = [1, 2, 3, 4]
l is m
False
l == m
True
```

Data Types :-

Integer :-

ex:-
```
a = 5
type (a)
<class 'int'>
```

float :-

ex:-
```
a = 3.6
a
3.6
type (a)
<class 'float'>
```

ex:-
```
a = 5e3
a
5000.0
type(a)
<class 'float'>
```

complex :-

ex:-
```
a = 3+4j
a
(3+4j)
type (a)
<class 'complex'>
```

ex:-
```
a = 3+4j
b = 5+2j
a+b
(8+6j)
a-b
(-2+2j)
a*b
(7+26j)
a/b
(0.7931034482758621+0.4827586206896552j)
```

ex:-
```
x = complex (5, 19)
x
(5+19j)
x.real
5.0
x.imag
19.0
```

Boolean data data type :-

```
a = True
type(a)
<class 'bool'>
```

None type :-

```
a = None
type(a)
<class 'None Type'>
```

Sequence data types :-

List :-

* list is a collection of element it may be homogeneous or heterogeneous.

* list is mutable.

ex:- l = [6, 8, 3, 5, 6, 78, 2]

```
l
[6, 8, 3, 5, 6, 78, 2]
type(l)
<class 'list'>
l[0]
6
l[1]
8
```

Tuple :-

Tuple is a collection of elements it may be homogeneous or heterogeneous.

-> Tuple is immutable.

ex:- t = (4, 8, 1, 2, 7)

```
type(t)
<class 'tuple'>
```

Set :-

-> Set is a collection of element.

-> Set is mutable, set elements are immutable. [List, set, dictionary can't be used as a set element].

-> Set doesn't allow duplicate elements.

-> Set doesn't maintain insertion order

-> It doesn't support indexing or subscript operation.

ex:-

S = {12, 56, 23, 12, 19, 18, 45, 32}

S

{32, 18, 19, 23, 56, 12, 45}

String :-

String is a collection of characters enclosed with in the codes.

-> String is immutable

ex:- S = "hello"

type(S)

<class 'str'>

Range :-

To generate sequence of values range will use.

range(lowerbound, upperbound, step)

range (lb, ub) default step = 1

range (ub) default

lb = 0

step = 1

Note :- python always exclude upper bound values.

when step value +ve lower limit should be less than upper limit otherwise, empty sequence will generate.

when step value -ve lower limit should be greater than upper limit otherwise empty sequence will generate.

ex:- list (range (1, 11))

[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

list (range(1, 11, 2))

[1, 3, 5, 7, 9]

list (range (2, 11, 2))

[2, 4, 6, 8, 10]

list (range (10, 0, -1))

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

list (range (5))

[0, 1, 2, 3, 4]

list (range (10, 5))

[]

list (range (10, 10))

[]

list (range (10, 20, -1))

[]

range is immutable

Mapping data type dictionary :-

Dictionary is a collection of key, value pairs.

key should be any immutable data type unique.

ex:- d = {'name' : 'Rani', 'age' : 21, 'depart' : 'CSE',
'marks' : 80}

d['marks']

Type conversion :-

converting from one data type to another data type is called type conversion.

To convert data into integer format.

int (float value)
int (decimal string)
int (numstring, base)

ex:- int (3.587)
3
int ("546")
546
int ("42a")
-> error
int ("45.678")
-> error
int (float ("45.678"))
45
int ("123", 10)
123

int ("123", 8)
83
int ("123", 16)
291
int ("1a", 16)
26
int ("10110", 2)
22
int ("0b101", 2)
5
int ("0x101", 16)
257
int ("00101", 8)
65

To convert to float

- float (45)
45.0
float ("45.65")
45.65

Decimal to octal

- oct(45)

'0055'

Decimal to hexadecimal

- hexa(55)

'0x37'

Decimal to binary

- bin(55)

'0b110111'

To convert data into string format

- str(a)

'123'

To convert numerical format to string

- str(45.67)

'45.67'

To know the ASCII value of a given character

ord('a')	ord('A')	ord(' ')	ord('0')
97	65	32	48
ord('z')	ord('2')	ord('+')	ord('9')
122	90	43	57

To convert ASCII value to character.

chr(98)

'b'

chr(52)

'4'

chr(69)

'E'

To know the absolute value abs = |-5|

abs(-56)

56.

eval() will consider one python statement as a argument it will return result of that statement.

```
->a = eval ("34")
  type (a)
  <class 'int'>
  a
  34
->a = eval ("34.56"]
  type (a)
  <class 'float'>
  a
  34.56
->a = eval ("4+4*2"]
  a
  12
->a = eval ("[3,4,5,6,7,8]")
   l = "[3,4,5,6,7,8]"
   type (l)
   <class 'str'>
   l = eval (l)
   type (l)
   <class 'list'>
   l
   [3,4,5,6,7,8]
```

To read inputs from the user at run time ~~for~~ input() will use it will take one string as a argument to provide reminder message to user what input they need to give

It will read entire line of data as a single string

ex:- a = input ("enter a value")
print (a, type (a))

o/p:- enter a value : 345
345 <class 'str'>

a = int (input ("enter a value"))
print (a, type (a))

o/p:- enter a value 345
345 <class 'int'>

read 3 integers line by line perform some of those 3 integers

i/p:- 2
3
4

2 3 4
9

a = int (input ('enter a value'))
b = int (input ('enter b value'))
c = int (input ('enter c value'))
print (a, b, c)
print (a+b+c)

Reading multiple inputs in same line :-

To read multiple inputs from the same line.

input() will return all the inputs as a single string, split() will divide it into list of tokens.

map() will map given function to every element in the sequence.

map (fun, seq)

a, b, c = map (int, input().split())

```
a,b,c = map(int, input().split())
print(a,b,c)
print(a+b+c)
```

o/p:- 3 4 5
3 4 5
12

By default print() will seperate all its arguments with spaces to change it sep will use.

ex:-
```
a,b,c = 5,8,24
print(a,b,c)
print(a,b,c, sep=",")
print(a,b,c, sep="\t")
print(a,b,c, sep="\n")
```

o/p:-
```
5 8 24
5,8,24
5	8	24
5
8
24
```

By default print() will end with new line to change it end will use.

ex:-
```
a,b,c = 5,8,24
print(a, end=" ")
print(b, end=" ")
print(c)
```

Formatting Integer, float:-

```
a = 43
b = 6
c = 4576
print("%05d" %(a))
print("%05d" %(b))
```

o/p: 00043
00006
04576
00043

Ex: Print ("%.05d" %.(c))
Print ("{:05d}".format(a))

a = 34.456456
Print ("%.2f" %.(a))
print ("%.07.2f" %.(a))

o/p: 34.46
0034.46

Operator precedence, associativity & expression evaluation

In a given expression if more than one operator is there which operator need to evaluate will decide by operator precedence.

If more than one operator having same precedence as per associativity it will evaluate

( ) —> L to R

unary minus —— ~, **, } —> R to L

*, /, //, % }
+, - } —> L to R
&, |, ^, <<, >> }

<, >, <=, >=, ==, !=, in, not in, is, is not —> L to R

not —> R to L

and, or —> L to R

= —> R to L

Ex: 3 + 5 * 2 - 5/3
3 + 10 - 5/3
3 + 10 - 1.6
13 - 1.6
11.4

2 * (3+4-2//5) - 6//4
2 * (3+4-0) - 6//4
2 * (7-0) - 6//4
2 * 7 - 6//4
14 - 6//4
14 - 1
13

3 ** 2 ** 4
3 ** 16
43046721

5.1 & 3 * (2+5/6) - 8 % 2
= 10.66

Display how many 200, 20, 5, 1 denominations need to given for the given amount

```
a = int(input(" enter a value "))
print (a//200)
a = a % 200
print (a//20)
a = a % 20
print (a//5)
a = a % 5
print (a//1)
```

```
a = int(input())
print (a//200) - 3
b = a % 200 - 186
print (b//20) - 9
c = b % 20
print (c//5)
d = c % 5
print (d//1)
```

o/p:- enter a value 786

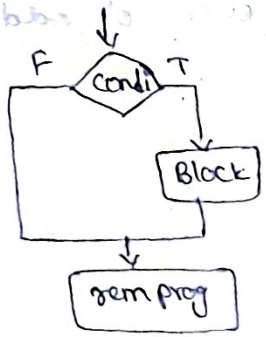
a	3
b	9
c	1
d	1

control statements :-

conditional or selection statement :-

Based on condition to execute a particular block or to skip

if :-




```
if cond :
    stm-1
    stm-2
    :
    stm-n
```

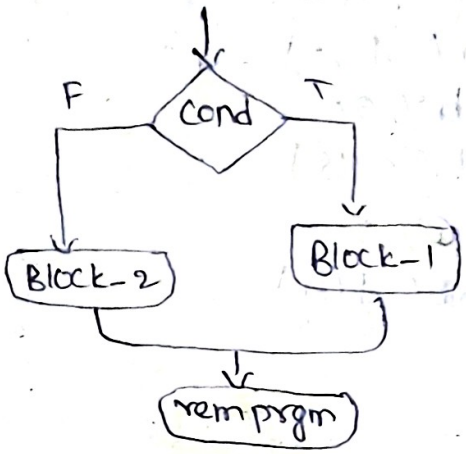
Display greatest among 2 numbers

ex:-

```
a = int(input())
b = int(input())
if a>b:
    print(a)
if b>=a:
    print(b)
print('end of pgm')
```

o/p:- 5
8
8
'end of pgm'

if-else:-




```
if condition
    stm-1
    stm-2
    :
    stm-n
else
    stm-1
    stm-2
    :
    stm-n
```

ex:-

```
a = int(input())
b = int(input())
if a>b:
    print(a)
else:
    print(b)
```

Check given number is even or odd

ex:-

```
n = int(input())
if n%2==0:
    print('even')
else:
    print('odd')
```

1. check given number is multiple of 3 or not.
2. check n is divisible by either 3 or 5.
3. check given number ending with either 4 or 7.
4. check given number is perfect square or not.

```
n = int (input ())
sqrt_n = int (n**0.5)
if sqrt_n * sqrt_n == n :
    print ('ps')
else :
    print ('not')
```

o/p:- 25
yes

5. By reading age as input check that person was eligible to vote or not.
6. By reading year as input check it is leap year or not.

```
Y = int (input ())
if (Y%.4 == 0 and Y%.100 != 0) or Y %. 400 == 0:
    print('yes')
else:
    print('NO')
```

1.
```
n = int(input())
if n %. 3 == 0:
    print ('yes')
else:
    print('NO')
```
o/p:- 7
NO

2.
```
n = int(input())
if n%.3 == 0 or n%.5 == 0 :
    print ('yes')
else :
    print ('NO')
```
o/p:- 15
yes

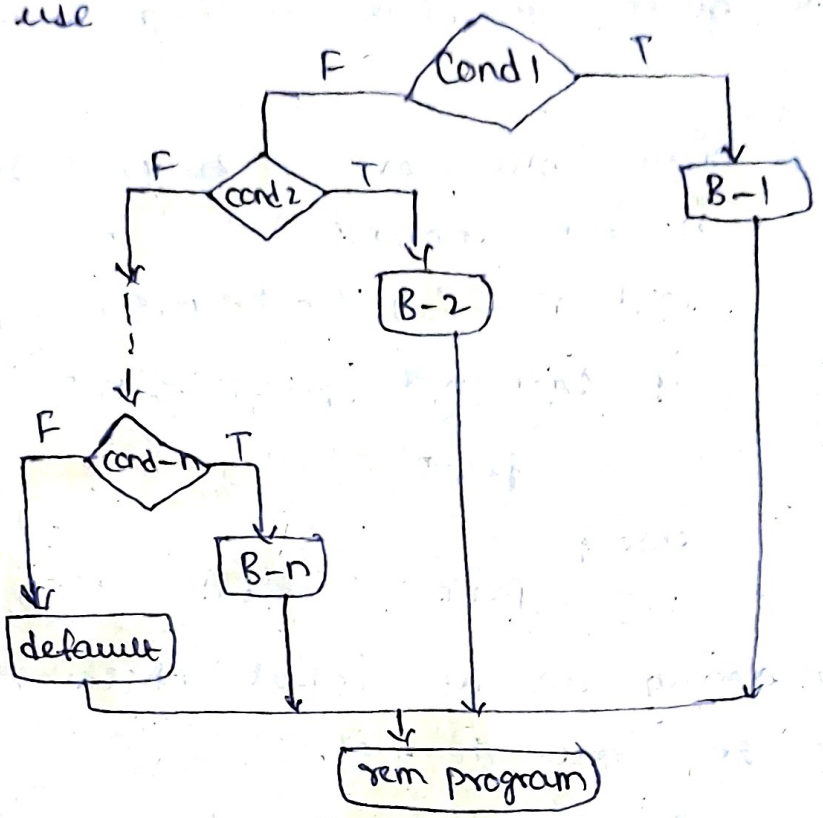
3.
```
n = int(input())
if n%.4 == 0 or n%.7 == 0
    print ('yes')
else
```
o/p:- 15.4
yes

5.
```
n = int(input()),
if n > 18 :
    print ('yes')
else:
    print('NO')
```
o/p:- 76
yes

elif ladder :-

Based on the condition to select any one block it will use

```
if cond1:
    stat-1
    :
    stat-n
elif cond2:
    stat-1
    :
    stat-n
else
    stm-1
    :
    stm-n
```



* Display greatest among 3 numbers.

```
a,b,c = map(int, input().split())
if a>b and a>c:
    print(a)
elif b>c:
    print(b)
else:
    print(c)
```

* check given number is positive, negative or zero.

```
n = int(input())
if n>0:
    print('+ve number')
elif n<0:
    print('-ve number')
else:
    print('zero')
```

*check given character is lower case or upper case or digit or space or special symbols.

```
ch = input ()
if ch >= 'a' and ch <= 'z':
    print ('lower case')
elif ch >= 'A' and ch >= 'Z':
    print ('upper case')
elif ch >= '0' and ch >= '9':
    print ('digit')
elif ch == ' ':
    print ('space')
else:
    print ('special symbol')
```

*Read the marks as a ip display the grade of a student.

```
m = int (input ())
if m >= 90:
    print ('O')
elif m >= 80:
    print ('E')
elif m >= 70:
    print ('A')
elif m >= 50:
    print ('B')
elif m >= 40:
    print ('C')
else:
    print ('F')
```

* calculate electricity bill by reading previous meter reading, current meter reading as a input based on following slab rates. 0-100 -> 2.5 Rs/unit, 101-500 -> 5.4 Rs/unit, 151-200 -> 7.2 Rs/unit, >200 -> 9.8 Rs/unit

```
Pm = int (input ())
cm = int (input ())
n = cm - pm
if n >= 0 and n <= 100:
    print (n * 2.5)
elif n >= 101 and n <= 150:
    print (n * 5.4)
elif n >= 151 and n <= 200:
    print (n * 7.2)
else:
    print (n * 9.8)
```

Nested if :-

```
if cond :
    if cond :
        stm-1
        :
        stm-n
    else :
        stm-1
        :
        stm-n
else :
    stm-1
    :
    stm-n
```

*Display greatest among 3 numbers without using logical operator.

```
a = int(input())
b = int(input())
c = int(input())
if a>b :
    if a>c :
        print(a)
    else :
        print(c)
elif b>c :
    print(b)
else :
    print(c)
```

* Sort given 3 numbers in ascending order

```
a = int(input())
b = int(input())
c = int(input())
if a>b and a>c :
```

```
    if b>c:
        print(c,b,a)
    else:
        print(b,c,a)
elif b>c:
    if a>c:
        print(c,a,b)
    else:
        print(a,c,b)
else:
    if a>b:
        print(b,a,c)
    else:
        print(a,b,c)
```

Match :-

Match is a alternative for else-if ladder based on choice value any one of the case will execute if choice doesn't match with any case label default case will execute

```
match choice:
    case ch1:
        stm-1
        =
        stm-n
    case ch2:
        stm-1
        =
        stm-n
    default
    choice case_:
        stm-1
        =
        stm-n
```

* check given number is even or odd

```
n = int (input())
match n%.2:
    case 0:
        print ("even")
    case 1:
        print ("odd")
```

* perform the arithmetic operations by reading operators as a choice.

```
a = int (input())
b = int (input())
no ch = input()
match ch:
    case '+':
        print (a+b)
    case '-':
        print (a-b)
    case '*':
        print (a*b)
    case '/':
        print (a/b)
    case '//':
        print (a//b)
    case '%':
        print (a%.b)
    case '**':
        print (a**b)
    case _:
        print ("enter valid operator")
```

Single line if-else :-

exp1 if and else exp2

* Display greatest among 2 numbers

```
a = 5
b = 7
print (a if a > b else b)
```

ex
```
n = 5
print ("even" if n%2 == 0 else "odd")
```

ex:- implementing grading of students using match

```
m = int (input())
match m:
    case m if m >= 90:
        print ("O")
    case m if m >= 60:
        print ("A")
    case m if m >= 40:
        print ("B")
    case _:
        print ("F")
```

looping or iterative statements :-

To execute some block of statements multiple times these statements will use.

for :-

```
for var in sequence:
    Stm-1
    :
    Stm-n
```

* Display 1-n numbers

```
n = int(input())
for i in range(1, n+1):
    print(i)
```

* Display even numbers b/w 1-n.

```
n = int(input())
for i in range(1, n+1):
    if i%2==0:
        print(i)
```

i	i%2	print(i)
1	1%2≠0	-
2	2%2==0	2
3	3%2!=0	-
4	4%2==0	4
⋮		
10	10%2==0	10

* Display alternate even numbers b/w 1-n.

```
n = int(input())
for i in range(1, n+1):
    if i%2==0 and i%4!=0:
        print(i)
```

o/p:
2
6
10

```
n = int(input())
for i in range(1, n+1):
    if i%4==0:
        print(i)
```

o/p: 4
8

* Display 1-n numbers that are divisible by either 3 or 5.

```
n = int(input())
for i in range(1, n+1):
    if i%3==0 or i%5==0:
        print(i)
```

* Display perfect squares b/w 1-n.

```
n = int(input())
for i in range(1, n+1):
    sq_i = int(i ** 0.5)
    if sq_i * sq_i == i:
        print(i)
```

* write a program to calculate sum of 1-n numbers.

```
n = int(input())
s = 0
for i in range(1, n+1):
    s = s + i
print(s)
```

n=5, i	s = s+i
	0
1	s = 0+1 = 1
2	s = 1+2 = 3
3	s = 3+3 = 6
4	s = 6+4 = 10
5	s = 10+5 = 15

* Display product of 1-n numbers. (factorial of a given number).

```
n = int(input())
f = 1
for i in range(1, n+1):
    f = f * i
print(f)
```

n=5, i	f = f * i
	1
1	f = 1 * 1 = 1
2	f = 1 * 2 = 2
3	f = 2 * 3 = 6
4	f = 6 * 4 = 24
5	f = 24 * 5 = 120

* Display mathematical table for a given number

```
n = int(input())
for i in range(1, 11):
    print(f"{n} * {i} = {n*i}")
```

* Display first n terms in the fibonacci series.

```
a = -1
b = 1
for i in range(n):
    c = a + b
    print(c)
    a, b = b, c
```

n = 5

i	a	b	c = a + b
	-1	1	
0	1	0	-1 + 1 = 0
1	0	1	1 + 0 = 1
2	1	1	0 + 1 = 1
3	1	2	1 + 1 = 2
4	2	3	1 + 2 = 3

* In a sequence first 3 terms are a=5, b=8, c=9 next term was defined as previous term + its previous two terms difference display nth term in this sequence (n > 3)

```
a = 5
b = 8
c = 9
for i in range (n-3):
    d = c + b - a
    a, b, c = b, c, d
print(d)
```

a = 5, b = 8, c = 9
d = 9 + 3 = 12
a, b, c = 8, 9, 12
d = 12 + 1 = 13
a, b, c = 9, 12, 13
d = 13 + 3 = 16
a, b, c = 12, 13, 16
d = 16 + 1 = 17
a, b, c = 13, 16, 17

* Display factors for a given number.

```
n = int(input())
for i in range(1, n+1):
    if n%i == 0
        print(i)
```

o/p:- 8
1
2
4
8

n = 8,

i	n%i == 0	print(i)
1	8%1 == 0	1
2	8%2 == 0	2
3	8%3 == 0	-
4	8%4 == 0	4
5	8%5 == 0	-
6	8%6 == 0	-
7	8%7 == 0	-
8	8%8 == 0	8

* Count number of factors for a given number.

```
n = int(input())
c = 0
for i in range(1, n+1):
    if n%i == 0:
        c+=1
print(c)
```

n = 6

i	n%i	c+=1 (c=c+1)
1	0	1
2	0	2
3	0	3
4		
5		
6	0	4

* check given number is perfect or abundant or deficient.

sum of factors == n perfect number

sum of factors < n deficient number

sum of factors > n abundant number

```
n = int(input())
s = 0
for i in range(1,n):
    if n%i == 0:
        s+=i
if s==n:
    print("perfect")
elif s<n:
    print("deficient")
else:
    print("abundant")
```

n = 6,

i	n%i	s=s+i
1	0	1
2	0	3
3	0	6
4	-	-
5	-	-

break :-

Break is used to terminate the current loop before ending the sequence.

```
for vm in x:
    if cond:
        break
```

ex:
```
for i in range(10,20):
    if i%3==0:
        print(i)
        break
```
O/P: 12

*
```
for i in range(10,20):
    print(i)
    if i%3==0:
        break
```
O/P: 10
11
12

*
```
for i in range(10,20):
    if i%3==0:
        break
    print(i)
```
O/P: 10
11

*
```
for i in range(11,25):
    if i%5==0 or i%7==0:
        break
    print(i)
```
O/P: 11
12
13

using flag

*
```
f=false
for i in range(11,45):
    if i%5==0 and i%7==0:
        f=true
        break
if f:
    print("loop ended by break")
else:
    print("loop ended by seq")
```

using for-else

*
```
for i in range(16,19):
    if i%5==0 and i%7==0:
        print("loop ended by break")
        break
else:
    print("loop ended by seq")
```

* To identify whether event was happened within the loop or to identify loop was ending with sequence or not flag variable is false.

* Display first perfect square b/w m-n if not available print -1.

using flag

```
m=int(input())
n=int(input())
f=true
for i in range(m,n+1):
    sqr_i=int(i**0.5)
    if sqr_i*sqr_i==i:
        f=false
```

```
print(i)          o/p: 10      o/p: 5
break                  25           6
                       16           -1
if f:
    print(-1)
```

* python uses for-else to avoid the flag variable.
* If loop is terminated due to break else part will not execute.
* If loop is terminated due to end of the sequence else part will execute.

ex:
```
m = int(input())
n = int(input())
for i in range(m, n+1):
    sqr_i = int(i**0.5)
    if sqr_i * sqr_i == i:
        print(i)
        break
else:
    print(-1)
```

* To check whether its prime or not, , for-else.

```
n = int(input())
if n<2:
    print("not prime")
else:
    for i in range(2,n):
        if n%i==0:
            print("not a prime")
            break
    else:
        print("prime")
```

n=5, 5<2 ×

i	n%i==0
2	5%2==0 ×
3	5%3==0 ×
4	5%4==0 ×

prime

n=9,

i	n%i==0
2	9%2==0 ×
3	9%3==0 ✓

not a prime

while :-

while is a event control loop it will execute block of statements until given condition becom false.

Syntax :-

```
while cond :
    stm-1
    stm-2
    :
    stm-n
```

* Display 1-n numbers using while

```
n = int(input())
i = 1
while i <= n :
    print(i)
    i = i + 1
```

n=5,	i<=n	i	i=i+1
i=1,	1<=5	1	2
i=2,	2<=5	2	3
i=3,	3<=5	3	4
i=4,	4<=5	4	5
i=5,	5<=5	5	6

* fibonacci series display n^{th} term using while

```
a = -1
b = 1
n = int(input())
i = 1
while i <= n :
    c = a + b
    a, b = b, c
    i = i + 1
print(c)
```

* check given number is prime or not using while

```
n = int(input())
if n < 2 :
    print("not a prime")
else :
    i = 2
```

n = 5,

i	i<n	n%i == 0
2	2<5	5%2 == 0 X
3	3<5	5%3 == 0 X
4	4<5	5%4 == 0 X
5	5<5 X	

prime

```
while i<n :
    if n%i==0 :
        print("not prime")
        i = i+1
        break
else :
    print("prime")
```

n=12,

i	i<n	n%i==0
2	2<12 ✓	12%2==0 ✓

not prime

* Display perfect squares b/w 1-n using while

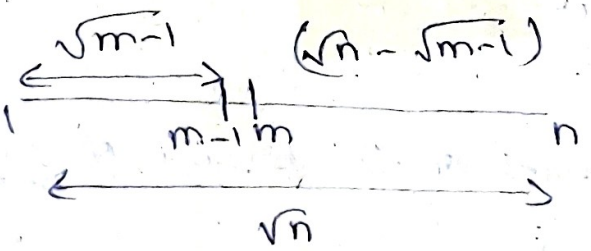
```
n= int(input())
i=1
c=0
while i<=n :
    sqrt_i = int(i**0.5)
    if sqrt_i * sqrt_i == i :
        c=c+1
        #print(i)
    i=i+1
print(c)
```

*

```
n= int(input())
print(int(n**0.5))
```

o/p: 9 16 b/w 1-n
3 4 int(√n)
int(n**0.5)

* how many perfect squares b/w m-n




```
m,n = map(int, input().split())
print(int((n**0.5)-((m**0.5)-1)))
```

* Count how many digits are there in a given number.

```
n = int(input())
c = 0
while n != 0:
    c = c + 1
    n = n // 10
print(c)
```

o/p: 542
3

n = 542

	c	n = n//10
n != 0	0	
542 != 0 ✓	0+1 = 1	n = 542//10 = 54
54 != 0 ✓	1+1 = 2	n = 54//10 = 5
5 != 0 ✓	2+1 = 3	n = 5//10 = 0
0 != 0 ✗		

* calculate sum of individual digits of a given number.

```
n = int(input())
s = 0
while n != 0:
    d = n % 10
    s = s + d
    n = n // 10
print(s)
```

o/p: 452
11

n = 452

	d = n % 10	s = s + d	n = n//10
n != 0			
452 != 0 ✓	d = 452 % 10 = 2	0+2 = 2	452//10 = 45
45 != 0 ✓	d = 45 % 10 = 5	2+5 = 7	45//10 = 4
4 != 0	d = 4 % 10 = 4	7+4 = 11	4//10 = 0

* check given number is palindrome or not.

```
n = int(input())
t = n
r = 0
while n != 0:
    d = n % 10
    r = r * 10 + d
    n = n // 10
if t == r:
    print("palindrome")
else:
    print("not a palindrome")
```

n = 24
t = n
r = 0
24 != 0
d = 24 % 10 = 4
r = 0 * 10 + 4 = 4
n = 24 // 10 = 2
2 != 0
d = 2 % 10 = 2
r = 4 * 10 + 2 = 42
n = 2 // 10 = 0
0 != 0 ✗
t == r ?
42 == 24 ✗

n = 22,
t = 22,
r = 0,
22 != 0 ✓
d = 22 % 10 = 2
r = 2
n = 22//10 = 2
2 != 0 ✓
d = 2 % 10 = 2
r = 2 * 10 + 2 = 22
n = 2 // 10 = 0
0 != 0 ✗
t == r ?
22 == 22 ✓

* Count how many times given digit repeated with in a given number.

```
n = int(input())
k = int(input())
c = 0
while n!=0:
    d = n%10
    n = n//10
    if d==k:
        c = c+1
print(c)
```

o/p:- 123456754
5
2

* check given number contain either 5 or 7.

```
n = int(input())
while n!=0:
    d = n%10
    n = n//10
    if d==5 or d==7:
        print('yes')
        break
else
    print('NO')
```

* Display maximum digit from a given number.

```
n = int(input())
max = 0
while n!=0:
    d = n%10
    n = n//10
    if d > max:
        max = d
print(max)
```

* write a program to calculate GCD of 2 numbers.

```
a = int(input())
b = int(input())
while b!=0:
    r = a%b
    a, b = b, r
print(a)
```

o/p: ~~27~~ 26
~~26~~ 27
~~1~~ 1

~~a=9, b=26~~

~~26) 27 (1~~
~~26~~
~~1) 27 (26~~
~~27) 26 (~~

a=12 b=18

```
      a
b-18) 12 (0
      0
      a
r 12) 18 (1
      12
      a
b  6) 12 (2
      12
      a
rb 0) 6 (
```

x=24, b=15

b!=0	r = a%b	a, b = b, r
15!=0	r = 24%15 = 9	a, b = 15, 9
9!=0	r = 15%9 = 6	a, b = 9, 6
6!=0	r = 9%6 = 3	a, b = 6, 3
3!=0	r = 6%3 = 0	a, b = 3, 0
0!=0 ×		

---

* Read unspecified number of integers, count how many +ve, -ve numbers entered by a user when user i/p was '0' stop reading the inputs.

```
pc = nc = 0
while True:
    n = int(input())
    if n > 0:
        pc += 1
    elif n < 0:
        nc += 1
    else:
        break
print(pc, nc)
```

IMP

- Break is used to terminate the current loop.
- Continue is used to terminate the current iteration.
- pass is used to define empty blocks.

Display 1-n numbers except that are divisible by 3.

```
n = int(input())
for i in range(1, n+1):
    if i%3==0:
        continue
    print(i)
```

	n=5
i	i%3==0
1	1%3==0 ✗
2	2%3==0 ✗
3	3%3==0 ✓
4	4%3==0 ✗
5	5%3==0 ✗

using pass

```
n = int(input())
for i in range(1, n+1):
    if i%3==0:
        pass
    else:
        print(i)
```

Nested loops :-

for every ~~outer~~ iteration of outer loop. nested loop will execute completely.

```
for i in range(1, 5):
    for j in range(1, 3):
        print(i, j)
    print("end of j loop")
print("end of i loop")
```

```
n = int(input())
for i in range(1, n+1):
    for j in range(1, n+1):
        print(j, end="")
    print()
```

O/P:- 5

1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5
1 2 3 4 5.

```
n = int(input())
for i in range(1, n+1):
    for j in range(1, n+1):
        print(i, end = " ")
    print()
```

O/P:- 5

1 1 1 1 1
2 2 2 2 2
3 3 3 3 3
4 4 4 4 4
5 5 5 5 5

```
n = int(input())
for i range(1, n+1):
    for j in range(1, n+1):
        print("*", end = " ")
    print()
```

O/P:- 5

* * * * *
* * * * *
* * * * *
* * * * *
* * * * *

* Display prime numbers b/w 1-n.

```
n=int(input())
for i in range(2,n+1):
    for j in range(2,i):
        if i%j==0:
            break
    else:
        print(i)
```

i	j	i%j==0	Prime
2	r(2,2)	-	2
3	r(2,3)	3%2==0 x	3
4	r(2,4) 2,3	4%2==0 ✓	-
5	r(2,5)	5%2==0 x	
	3	5%3==0 x	
	4	5%4==0 x	
6	r(2,6) 2,3,4,5	6%2==0 ✓	-

* Display prime factors for a given number.

```
n=int(input())
for i in range(2,n+1):
    if n%i==0:
        for j in range(2,i):
            if i%j==0:
                break
        else:
            print(i)
```

* Display first n prime numbers.

```
n=int(input())
i=2
c=0
while True:
    for j in range(2,i):
        if i%j==0:
            break
    else:
        c=c+1
        print(i)
        break
        if c==n:
            break
    i=i+1
```

* sum of individual digits of a given number until become a single digit.

```
n = int(input())
while n>9:
    s=0
    while n!=0:
        s = s + n%10
        n = n//10
    n=s
print(n)
```

n = 9947
s = 29
n = 29
s = 11
n = 11
s = 2

(or)

```
n = int(input())
while n>9:
r = n%9
print(9 if r==0 else r)
```

```
* n = int(input())
  for i in range(1, n+1):
      for j in range(1, i+1):
          print(j, end=" ")
      print()
```

o/p: 5

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

```
* n = int(input())
  for i in range(1, n+1):
      print(" " * (n-i), end=" ")
      for j in range(1, i+1):
          print(j, end=" ")
      print()
```

o/p: 5

```
    1
   1 2
  1 2 3
 1 2 3 4
1 2 3 4 5
```

```
n = int(input())
for i in range(1, n+1):
    print(" " * (n-i), end="")
    for j in range(n, n-i, -1):
        print(j, end=" ")
    print()
```

o/p: 5

```
    5
  5 4
5 4 3
5 4 3 2
5 4 3 2 1
```

```
n = int(input())
for i in range(1, n+1):
    print(" " * (n-i), end="")
    for j in range(1, i+1):
        print(j, end=" ")
    for j in range(i-1, 0, -1):
        print(j, end=" ")
    print()
for i in range(n-1, 0, -1):
    print(" " * (n-i), end="")
    for j in range(1, i+1):
        print(j, end=" ")
    for j in range(i-1, 0, -1):
        print(j, end=" ")
    print()
```

o/p: 5

```
    1
  1 2 1
1 2 3 2 1
1 2 3 4 3 2 1
1 2 3 4 5 4 3 2 1
1 2 3 4 3 2 1
  1 2 3 2 1
    1 2 1
      1
```

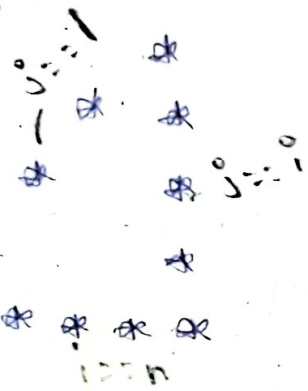
```
n = int(input())
for i in range(1,n+1):
    for j in range(1,n+1):
        print(i if i<j else j, end=" ")
    for j in range(n-1,0,-1):
        print(i if i<j else j, end=" ")
    print()
for i in range(n-1,0,-1):
    for j in range(1,n+1):
        print(i if i<j else j, end=" ")
    for j in range(n-1,0,-1):
        print(i if i<j else j, end=" ")
    print()
```

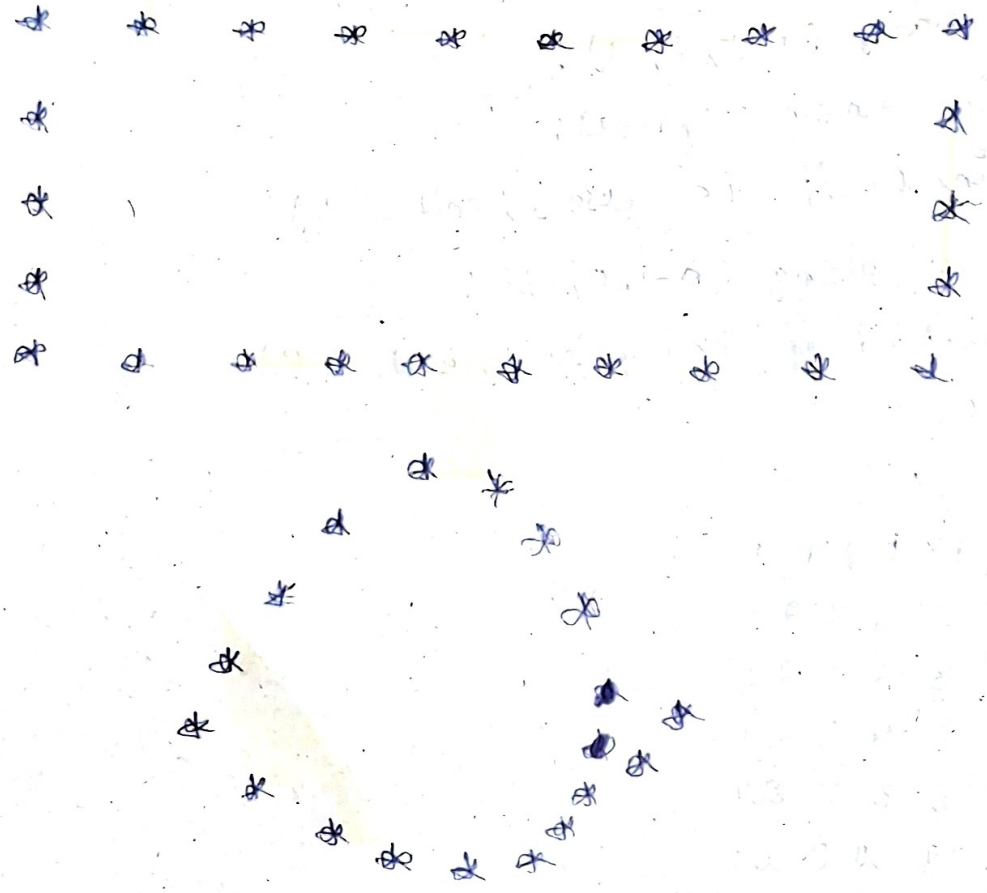
o/p: 4

```
1 1 1 1 1 1 1
1 2 2 2 2 2 1
1 2 3 3 3 2 1
1 2 3 4 3 2 1
1 2 3 3 3 2 1
1 2 2 2 2 2 1
1 1 1 1 1 1 1
```

```
n = int(input())
for i in range(1, n+1):
    print(" "*(n-i), end=" ")
    for j in range(1, i+1):
        if j==1 or j==i or i==n:
            print("*", end=" ")
        else:
            print(" ", end=" ")
    print()
```

O/P: 5





```
n = int(input())
for i in range(1, n+1):
    print(" " * (n-i), end=" ")
    for j in range(1, 2*i):
        if j==1 or j==2*i-1:
            print('*', end=" ")
        else:
            print(' ', end=" ")
    print()
for i in range(n-1, 0, -1):
    print(" " * (n-i), end=" ")
    for j in range(1, 2*i):
        if j==1 or j==2*i-1:
            print('*', end=" ")
        else:
            print(' ', end=" ")
    print()
```

o/p:

(or)

```
for j in range(1, i+1):
    print('*' if j==1 else ' ', end=" ")
for j in range(i-1, 0, -1):
    print('*' if j==1 else ' ', end=" ")
print()
for i in range(n-1, 0, -1):
    print(" " * (n-i), end=" ")
    for j in range(1, i+1):
        print('*' if j==1 else ' ', end=" ")
    for j in range(i-1, 0, -1):
        print('*' if j==1 else ' ', end=" ")
```

```
n = int(input())
for i in range(1, n+1):
    for j in range(1, n+1):
        if i==1 or j==1 or i==n:
            print('*', end=" ")
        else
            print(' ', end=" ")
    for j in range(n-1, 0, -1):
        if i==1 or j==1 or i==n:
            print('*', end=" ")
        else:
            print(' ', end=" ")
    print()
```

o/p: 5

```
* * * * * * * * *
*               *
*               *
*               *
* * * * * * * * *
```

```
n = int(input())
for i in range(1,n+1):
    for j in range(1,n+1):
        print(i+(j-1)*n, end=" ")
    print()
```

o/p: 7

```
1  8  15 22 29 36 u
2  9  16 23 30 37 u
3  10 17 24 31 38 u
4  11 18 25 32 39 u
5  12 19 26 33 40 u
6  13 20 27 34 41 u
7  14 21 28 35 42 u
```

* n = int(input())
c = 1
for i in range(1, n+1):
    for j in range(1, i+1):
        print("%2d" %(c), end=" ")
        c = c+1
    print()

o/p:

1
2 3
4 5 6
7 8 9 10
11 12 13 14 15

* n = int(input())
for i in range(1, n+1):
    for j in range(1, i+1):
        print(chr(96+j), end=" ")
    print()

(96+1) -> a
(96+2) -> b
(96+3) -> c
:

o/p:-

a
a b
a b c
a b c d
a b c d e

list :-

list is a collection of elements it may be homogeneous or heterogeneous.

list is mutable.

It supports both +ve, -ve indexing. Always index starts from 'zero' onwards.

ex:

-6	-5	-4	-3	-2	-1
3	5	12	7	15	19
0	1	2	3	4	5

l[2] or l[-4] -> 12

l[5] or l[-1] -> 19

To know the number of elements in a given list. len() will use.

ex: l = [5, 8, 1, 2]
print(len(l))

o/p: 4

Accessing data from a list :

ex: l = [5, 8, 1, 2, 5, 9, 8]

```
for i in range(len(l)):
    print(l[i])
```

o/p: 5
8
1
2
5
9
8

* l = [5, 8, 1, 2, 5, 9, 8]

```
for el in l:
    print(el)
```

o/p: 5
8
1
2
5
9
8

* l = [5, 8, 1, 2, 5, 9, 8]

```
print(*l)
print(*l, sep="\t")
print(*l, sep="\n")
```

o/p: 5 8 1 2 5 9 8
5 8 1 2 5 9 8

5
8
1
2
5
9
8

- To add element ~~x in to a list~~ at end position append() will use.

* l = [5, 4, 56, 2, 3]

```
l.append(12)
print(l)
```

o/p: [5, 4, 56, 2, 3, 12]

- Creating list if inputs are line by line

* n = int(input())

```
l = []
for i in range(n):
    l.append(int(input()))
print(l)
```

o/p: 5
8
1
3
4
9
[8, 1, 3, 4, 9]

List comprehension : l = [exp for v in seq]

```
n = int(input())
l = [int(input()) for i in range(n)]
print(l)
```

op:- 4
9
8
5
2
[9, 8, 5, 2]

If inputs are in same line :-

```
* l = list(map(int, input().split()))
  print(l)
```

op:- 8 35 70 434 5869
[8, 35, 70, 434, 5869]

```
* l = []
  for i in input().split():
      l.append(int(i))
  print(l)
```

op:- 4 56 2 35 80
[4, 56, 2, 35, 80]

```
* l = [int(i) for i in input().split()]
  print(l)
```

op:- 34 56 85 90 80
[34, 56, 85, 90, 80]

* Display index position of given element within a list if not available print -1 :(linear search).

```
l = list(map(int, input().split()))
k = int(input())
for i in range(len(l)):
    if k == l[i]:
        print(i)
        break
else:
    print(-1)
```

k = 7

3	8	2	9	4	7	6
0	1	2	3	4	5	6

i i i i i i

4 < < < < <

=> 5

* count how many times given element repeated within a given list.

```
def rep():
    l = list(map(int, input().split()))
    k = int(input())
    c = 0
    for i in range(len(l)):
        if l[i] == k:
            c += 1
    return c
print(c)
print(rep())
```

~~k=7~~

o/p: 6 7 5 3 5 3 8

0 1 2 3 4 5 6

c = 0 1 2

5

2

* Display sum of list of elements.

```
l = list(map(int, input().split()))
s = 0
for i in l:
    s = s + i
print(s)
```

o/p: 5 7 4 1 2

```
def sum(l):
    l=
    s=0
    for i in l:
        s=s+i
    return s
print(sum(s))
```

* count how many even numbers are there in a given list.

```
l = list (map( int, input().split()))
c=0
for i in l:
    if i%2==0:
        c=c+1
print(c)
```

O/P: 2 5 8 7 9

2

* from the given list to create two lists one contain even numbers another contain odd numbers.

```
l = list (map( int, input().split()))
even = []
odd = []
for i in l:
    if i%2==0:
        even.append(i)
    else:
        odd.append(i)
print("even:", even)
print("odd:", odd)
```

O/P: 23 43 24 76 46 73

even: [24, 76, 46]

odd: [23, 43, 73]

* find minimum, maximum value from a given list.

```
l=list(map(int, input().split()))
min = max = l[0]
for el in l:
    if el < min:
        min = el
    elif el > max:
        max = el
print(min, max)
```

3	8	2	9	7	6	4
0	1	2	3	4	5	6

min = 3 2

max = 3 8 9

3<3 x 8<3 x

3>3 x 8>3 ✓

2<3 ✓ 9<2 x

2>8 x 9>8 ✓ 4<2 x

7<2 x 6<2 x 4>9 x

7>9 x 6>9 x

min = 2

max = 9

* check given list was increasing order or not.

```
l = list(map(int, input().split()))
for i in range(len(l)-1):
    if l[i] > l[i+1]:
        print("no")
        break
else:
    print("yes")
```

O/P:- 8 12 20 31
yes

Adding element into a list:-

To add element at end position [l.append(eleme

* l = [5, 7, 2, 9, 8]
  l.append(3)
  print(l)

O/P: [5, 7, 2, 9, 8, 3]

To add element at specific ~~end~~ index position or beginning [l.insert(index, element)]

* Note:- If index position is invalid not exist by default the element will be added at end position.

* l = [4, 6, 2, 5]
  l.insert(2, 8)
  l.insert(0, 15)
  l.insert(10, 37)
  print(l)

O/P: [15, 4, 0, 18, 2, 5, 37]

* To update data at specific index

`l[index] = value`

```
* l = [5, 8, 2, 4, 9]
  l[1] = 21
  print(l)
```

O/P: [5, 21, 2, 4, 9]

Deleting element :-

- To delete data from ending `l.pop()`. It will return deleted element.

```
* l = [6, 9, 12, 54, 6, 8]
  print(l.pop())
  print(l)
```

O/P: 8
[6, 9, 12, 54, 6]

- To delete data from specific index or beginning. `l.pop(index)`

- If index was not exist it will throw error.

```
* l = [6, 9, 12, 54, 6, 8]
  print(l.pop(3))
  print(l.pop(0))
  print(l)
```

O/P: 54
6
[9, 12, 6, 8]

- To remove specific element [l.remove(element)]
- If given element was not available in the list it will throw error.

* l = [6, 9, 12, 54, 6, 8]
  l.remove(12)
  print(l)

o/p: [6, 9, 54, 6, 8]

- To delete object from the memory

[del object]

* a = 5
  del a
  #: print(a) --> error

* l = [4, 8, 1, 9, 7]
  del l[2]
  print(l)

o/p: [4, 8, 9, 7]

Searching

- To know given element was available in the list or not membership operator will use.

* l = [5, 8, 2, 4, 9]
  print [4 in l]
  print [14 in l]

o/p: True
  False

- To know the index position of the given element. `l.index(element, start ind, end ind)` (end ind → len(l); start ind → 0; end ind → optional)

* - If the given element was not available in the list it through error, if available it will return first occurance index position.

```
l = [6, 9, 12, 18, 23, 9, 14]
print(l.index(18))
# print(l.index(180)) --> error
print(l.index(9))
print(l.index(9,4))
# print(l.index(9,2,5)) --> error 9 not in 2,3,4
```

o/p: 3 (index)
1
5

- To count how many times given element repeated in a list. `l.count(element)`

```
l = [5,8,2,3,4,5,3,4,3,2,3]
print(l.count(3))
print(l.count(2))
print(l.count(13))
```

o/p: 4
2
0

* reverse the given list elements.

```
l = list(map(int, input().split()))
r = []
for i in range(len(l)-1, -1, -1):
    r.append(l[i])
print(r)
```

* remove shift all zeros to ending of a list.

```
l = list(map(int, input().split()))
c = l.count(0)
while 0 in l:
    l.remove(0)
for i in range(c):
    l.append(0)
print(l)
```

o/p - 23, 0, 5, 0, 7, 0

) 3, 5, 7, 0, 0, 0.

* rotate k elements from left to right

```
l = list(map(int, input().split()))
k = int(input())
for i in range(k):
    l.append(l.pop(0))
print(l)
```

o/p:

12 4 6 7 8 23 15

[8, 23, 15, 12, 4, 6, 7]

* l =

```
n = len(l)
for i in range(n//2):
    t = l[i]
    l[i] = l[n-i-1]
    l[n-i-1] = t
print(l)
```

* l = list(map(int, input().split

```
for i in range(len(l)):
    if l[i] == 0:
        l.append(l[i])
        l.remove(0)
print(l)
```

* l = list(map(int, input().split())

```
k = int(input())
for i in range(k):
    l.append(l[0])
    l.pop(0)
print(l)
```

reverse program

```
l = list(map(int, input().split()))
i = 0
j = len(l) - 1
while i<j:
    l[i], l[j] = l[j], l[i]
    i = i+1
    j = j-1
print(l)
```

i/p: 8 2 7 4 1 8 9 8
8 9 3 1 4 7 2 5

[8, 9, 3, 1, 4, 7, 2, 5]

shift all zeros

```
l = list(map(int, input().split()))
i = 0
for j in range(len(l)):
    if l[j] != 0:
        l.insert(i, l.pop(j))
        i = i+1
print(l)
```

i/p: 4 -2 0 0 3 0 6

[4, -2, 3, 6, 0, 0, 0]

Spy

```
n = 0
s = 0
p = 1
while n!=0:
    d = n%10
    s = s+d
    p = p*d
    n = n//10
if s==p:
    print("S")
else:
    print("NS N")
```

Harshad

```
n = 0
s = 0
t = n
while n!=0:
    s = s+n%10
    n = n//10
if t%s==0:
    P("H")
else:
    P("NH")
```

Strong

```
n = 0
s = 0
t = n
while n!=0:
    i = 1
    d = n%10
    for i in range(1, d+1):
        f = f*i
    s = s+f
    n = n//10
if s==t:
    P("S")
else:
    P("N")
```

* Rotate k elements

```
l = list (map( int, input().split()))
k = int ( input ())
k = k % len (l)
for i in range(k):
    l.append (l.pop(0))
print (l)
```

O/P: 4 76 2 43 5 67 3 5
3
[43, 5, 67, 3, 5, 4, 76, 2]

Sorting :- sort will work only list.

* To organize data either increasing or decreasing order sort will use. `l.sort()` -> Ascending

`l.sort(reverse = True)` -> descending

* l = [3, 5, 4, 20, -3, 2, -9]

```
l.sort()
print (l)
l.sort (reverse = True)
print (l)
```

O/P: [-3, -9, 2, 3, 4, 5, 20]
[20, 5, 4, 3, 2, -9, -3]

- sort will modify the source list it doesn't written anything. sorted will return a list it doesn't modify source list it will work for any sequence data type.

* l = [5, 3, 213, 45, 67, 3, -3, 9]
print (sorted (l))
print (sorted (l, reverse = True))

O/P: [-3, 3, 3, 5, 9, 45, 67, 213]
[213, 67, 45, 9, 5, 3, 3, -3]

* l = [5, 3, 213, 45, 67, 3, -3, 9]
print (min (l))
print (max (l))
print (sum (l))

O/P: -3
213
342

- To merge or concate two lists

* l = [4, 7, 2, 5]
m = [8, 3, 1]
print (l+m)
l.extend (m)
print (l)

O/P: [4, 7, 2, 5, 8, 3, 1]
[4, 7, 2, 5, 8, 3, 1]

- To reverse a given list

* l = [4, 7, 2, 5]
print(l[::-1])
l.reverse()
print(l)

O/P: [5, 2, 7, 4]
[5, 2, 7, 4]

- If L is a given `l.clear()` will remove all the elements from a given list.

```
l = [4,7,2,43]
l.clear()
print(l)
```

o/p:- []

- If you want to compare two lists having same values ~~or~~ or not

```
l = [4,8,7,1,2]
m = [4,8,7,1,2]
t = [4,7,8,2,1]
print(l==m)
print(l==t)
```

o/p:- True
False

- Delete duplicate elements from a given list.
- To remove the duplicate elements list will be converted into set it doesn't maintain insertion order.

```
l = [5,8,2,4,6,4,5,6,3,5]
print(list(set(l)))
```

o/p:- [2,3,4,5,6,8]

```
l = list(map(int,input().split()))
r = []
for el in l:
    if el not in r:
        r.append(el)
print(r)
```

o/p:- 4 7 3 2 5 4 3 2 1
[4,7,3,2,5,1]

* l = list(map(int, input().split()))

```
i=0
while i < len(l)-1 :
    j = i+1
    while j < len(l):
        if l[i] == l[j] :
            del l[j]
        else
            j = j+1
    i = i+1
print(l)
```

3 7 5 2 3 7 3 5 2 7

3 7 5 2 3 5 2 7

3 7 5 2 5 2

3 7 5 2 2

[3,7,5,2]

- A list contain numbers b/w m-n except 1 number display that missing number.

* l = list(map(int, input().split()))

```
for el in range(min(l), max(l)):
    if el not in l:
        print(el)
        break
```

O/P: 4, 5 8 6 9 10

7

- In a given list except 1 number remaining all the numbers were repeated more than one time display that lonely number.

* l = list(map(int, input().split()))

```
for el in set(l):
    if l.count(el) == 1:
        print(el)
        break
```

O/P: 3 6 1 2 2 3 3 2 2 3 1 1

6

- Count how many possible pairs are there in a given list such a way that sum of pair should be divisible by k.

```
l = list(map(int, input().split()))
k = int(input())
c = 0
for i in range(len(l)-1):
    for j in range(i,len(l)):
        if (l[i]+l[j])%k==0:
            c = c+1
print(c)
```

```
4 7 3 9 16
5
5
```

```
4 7 3 9 2 16
4 3 7 9 3 2
4 9 7 2 3 6
4 7 2 6
4 6
```

IMP

- Display second largest element from a given list.

```
l = list(map(int, input().split()))
for i in (set(l)):
    l.sort(l)
print(i)
```

(or)

```
l = list(map(int, input().split()))
m = max(l)
while m in l:
    l.remove(m)
print(max(l))
```

- Display which element repeated highest numbers of times in a given list.

```
l = list(map(int, input().split()))
max_count = 0
most_freq = None
for i in (set(l)):
    freq = l.count(i)
    if freq > max_count:
        max_count = freq
        most_freq = i
print(most_freq)
print(max_count)
```

* second largest element

```
l = list(map(int, input().split()))
l = sorted(set(l), reverse = True)
print(l[1])
```

* Breaking the records

```
n = int(input())
l = list(map(int, input().split()))
min = l[0]
max = l[0]
minc = 0
maxc = 0
for el in l:
    if el < min:
        min = el
        minc += 1
    elif el > max:
        max = el
        maxc += 1
print(maxc, minc)
```

* Birthday cake candles

```
n = int(input())
l = list(map(int, input().split()))
print(l.count(max(l)))
```

* min - max sum

```
l = list(map(int, input().split()))
s = sum(l)
print(s-max(l), s-min(l))
```

IMP

Binary search algorithm:

```
a = list(map(int, input().split()))
k = int(input())
l = 0
u = len(a) - 1
while l <= u:
    mid = (l+u)//2
    if a[mid] == k:
        print(mid)
        break
    elif k < a[mid]:
        u = mid - 1
    else:
        l = mid + 1
else:
    print(-1)
```

O/P: 2 6 6 6 6

2

2 5 7 8 12 19 2

k = 4
l = 0
u = 6
0 <= 6
mid = (0+6)//2 = 3
a[3] == k
8 == 4 x
4 < 8
u = 2
0 <= 2
mid = (0+2)//2 = 1
a[1] == 4
4 < 5
u = 0
0 <= 0
mid (0+0)//2 = 0
a[0] == 4
2
4 < 2
l = 0+1
l <= 0

Bubble sort :-

```
l = list(map(int, input().split()))
for i in range(len(l)-1):
    for j in range(i+1, len(l)):
        if l[j] < l[i]:
            l[i], l[j] = l[j], l[i]
print(l)
```

3 2 8 12 7 9 16 4 2

2 3 8 4 12 7 9 16 4 5

2 3 8 4 12 7 9 16 4 7

2 3 4 8 12 7 9 16 8 9

2 3 4 7 12 8 9 16 12 8

2 3 4 7 8 9 16 12 12 16

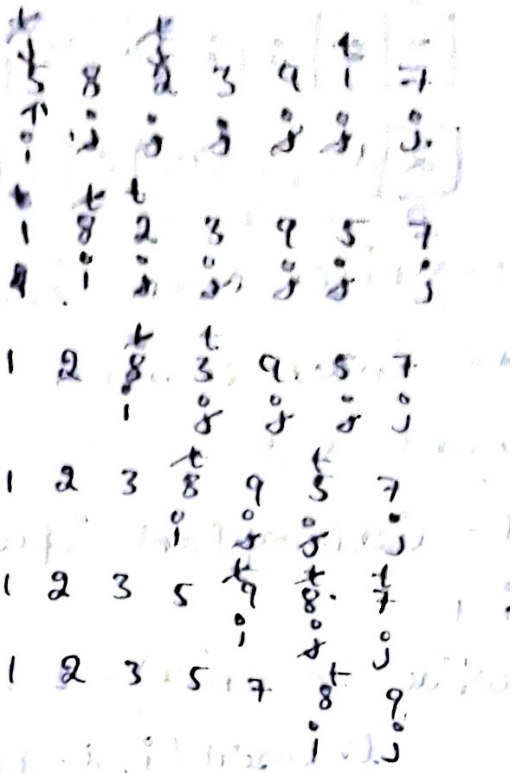
2 3 4 7 8 9 12 16

Selection sort :

```
l = list(map(int, input().split()))
for i in range(len(l)-1):
    t = i
    for j in range(i+1, len(l)):
        if l[j] < l[t]:
            t = j
    l[i], l[t] = l[t], l[i]
print(l)
```

o/p: 5 1 2 54 76 87 34 67

[1, 2, 5, 34, 54, 67, 76, 87]

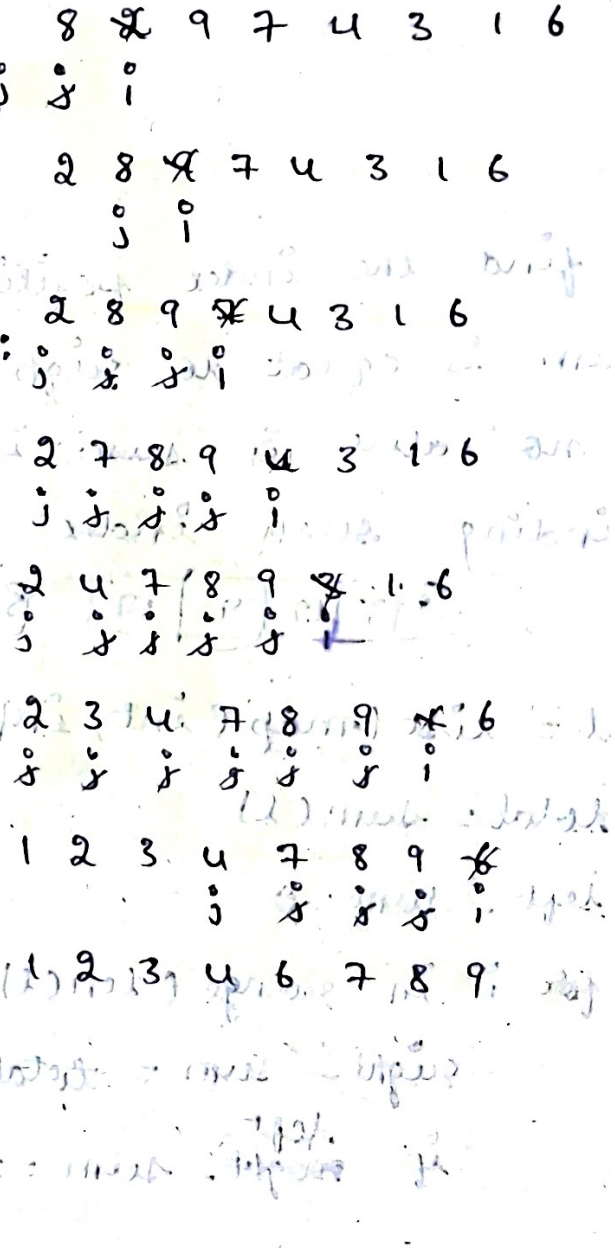


Insertion sort :-

```
l = list(map(int, input().split()))
for i in range(1, len(l)):
    j = i-1
    while j != -1 and l[j] > l[i]:
        j = j-1
    l.insert(j+1, l.pop(i))
print(l)
```

o/p: 8 2 9 7 4 3 1 6

[1, 2, 3, 4, 6, 7, 8, 9]



*

0	1	2	3	4	5	6
2	5	6	8	9	12	15

0	1	2	3	4	5	6
2	15	5	12	6	9	8

Organise even index elements in increasing order, odd index elements in decreasing order for a sorted list.

```
l = list(map(int, input().split()))
i=1
while i< len(l)-1:
    l.insert(i, l.pop())
    i=i+2
print(l)
```

2 5 6 8 9 12 15

2 15 5 6 8 9 12

2 15 5 12 6 9 8

* find the index position whose left side values sum is equal to right side values sum. if there is no value is sum is zero. If you're not finding such index position print(-1).

2	3	12	9	17

5	1	-2	4	9

```
l = list(map(int, input().split()))
total = sum(l)
left_sum = 0
for i in range(len(l)):
    right_sum = total - left_sum - l[i]
    if left_sum == right_sum
```

```
            print(i)
            break
        left_sum = left_sum + l[i]

else:
    print(-1)
```

ip:- 2 3 12 9 17 , 5 1 -2 -4 9

3 4

(or)

```
l = list(map(int, input().split()))
s = [0]
for i in range(len(l)):
    s.append(s[i]+l[i])
for i in range(len(l)):
    if s[i] == s[-1] - s[i+1]:
        print(i)
        break
else:
    print(-1)
```

5 8 -2 3 5 5

0 5 13 11 14 19 24

Slicing :-

l[startind : endind : step]

-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
4	8	12	54	35	78	1	9	16	15
0	1	2	3	4	5	6	7	8	9

```
l = [4,8,12,54,35,78,1,9,16,15]
print(l[::])
print(l[::-1])
print(l[:5]) # first 5 values
print(l[2:7]) # 2nd to 6th index
```

```
print(l[len(l)-3:]) # last 3 elements
print(l[-3:]) # last 3 elements
print(l[::2]) # alternate elements
print(l[7:4:-1]) # 7 ind to 5th ind
print(l[0:-3]) # except last 3 elements
print(l[-4::-1]) # except last 3 elements
print(l[7:2]) # invalid []
print(l[2:5:-1]) # invalid []
```

o/p:-

* index position find

```
l=list(map(int, input().split()))
for i in range(len(l)):
    if sum(l[:i]) == sum(l[i+1:]):
        print(i)
        break
else:
    print(-1)
```

* In a given list sort first k elements in ascending order and remaining elements in descending order.

* extract highest possible length sub list from a given list it contains unique elements.

```
l=list(map(int, input().split()))
f=False
for k in range(len(l),0,-1):
    for i in range(len(l)-k+1):
        x=l[i:i+k]
        if len(x)==len(set(x)):
            print(x)
            f=True
            break
    if f:
        break
```

* Replace every element in a list with the product of all other elements. [1,2,3,4] → [24,12,8,6]

```
l=list(map(int, input().split()))
p=1
for el in l:
    p=p*el
for i in range(len(l)):
    l[i]=p//l[i]
print(l)
```

1*24
24//1=24
24//2=12
24//3=8
24//4=6

* extract all k length sublist from a given list.

```
l = list(map(int, input().split()))
k = int(input())
for i in range(len(l)-k+1):
    print(l[i:i+k])
```

ip: 4 6 21 4 6 2 5 7 8

4

[4, 6, 21, 4]
[6, 21, 4, 6]
[21, 4, 6, 2]
[4, 6, 2, 5]
[6, 2, 5, 7]
[2, 5, 7, 8]

l = [5, 7, 3, 9, 7, 6]

[5,7,3] l[0:3]
[7,3,9] l[1:4]
[3,9,7] l[2:5]
[9,7,6] l[3:6]
[7,6,7] l[4:7]
[6,7,6] l[5:8]

i -> 0 -> len(l) - k + 1

* display all possible sublists from a given list.

```
l = list(map(int, input().split()))
for k in range(len(l), 0, -1):
    for i in range(len(l)-k+1):
        print(l[i:i+k])
```

Aliasing :-

For the same object giving the other name is called aliasing.

If we do modification with one name it will reflect to other name also.

```
* l = [5, 7, 3, 9, 12]
  m = l
  m[2] = 18
  print(l)
  print(m)
```

op: [5, 7, 18, 9, 12]
[5, 7, 18, 9, 12]

cloning :-

creating a copy of adjusting object is called cloning. If we do the modification on one object it doesn't reflect to other object.

```
* l = [3,7,8,1,7,4]
  # m = list(l)
  # m = l[:]
  m = l.copy()
  m[2] = 67
  print(l)
  print(m)
```

o/p: [3,7,8,1,7,4]
[3,7,67,1,7,4]

Nested - list :-

```
* l = [[5,8,9], [4,9,2,1], [6,9,3,5,1], [6,9]]
      0 1 2     0 1 2 3     0 1 2 3 4     0 1
  print(l[0])
  print(l[1])
  print(l[0][2])
```

o/p: [5,8,9]
[4,9,2,1]
9

```
* for i in range(len(l)):
      for j in range(len(l[i])):
          print(l[i][j], end=" ")
      print()
```

o/p: 5 8 9
4 9 2 1
6 9 3 5 1
6 9

```
* for row in l:
      for el in row:
          print(el, end=" ")
      print()
```

o/p: 5 8 9
4 9 2 1
6 9 3 5 1
6 9

```
* for row in l:
    print(*row)
```

```
o/p: 5 8 9
     4 9 2 1
     6 9 3 5 1
     6 9
```

creating Nested list with same no-of cols for every row :-

```
* r= int(input())
  c = int(input())
  l = []
  for i in range(r):
      x = []
      for j in range(c):
          x.append(int(input()))
      l.append(x)
  print(l)
```

```
o/p: 2
     3
     1
     2
     3
     4
     5
     6
[[1,2,3],[4,5,6]]
```

* creating nested list with different no. of cols in each row :

```
r = int(input())
l = []
for i in range(r):
    c = int(input(f"enter {i+1} row no. of cols"))
    x = []
    for j in range(c):
        x.append(int(input()))
    l.append(x)
print(l)
```

```
o/p: 2
enter 1 row 2
3
4
enter 2 row 3
4
5
6
[[3,4],[4,5,6]]
```

* By using list comprehension

```
r= int (input())
c = int (input())
l = [[int (input()) for j in range(c)] for i in range(r)]
print (l)
```

o/p: 2
4 5
6 7
8 9
[[4,5],[6,7],[8,9]]

* If inputs are row by row

```
r= int(input())
l = []
for i in range(r):
    l.append( list(map(int, input().split())))
print(l)
```

o/p: 3
2 3 4
1 2 3 4 5
4 8 6 2
[2,3,4] [1,2,3,4,5] [4,8,6,2]

* creating a list with default values with specified size.

```
n=5
l=[0]*n
print(l)
for i in range(n):
    l[i]= int (input())
print(l)
```

o/p: [0,0,0,0,0]
5
7
8
2
5
[5,7,8,2,5]

* l = [[0]*3 for i in range(5)]
print(l)
l[0][0]=12
print(l)

o/p: [[0,0,0], [0,0,0], [0,0,0], [0,0,0], [0,0,0]]

[[12,0,0], [0,0,0], [0,0,0], [0,0,0], [0,0,0]]

* min, max from a given nested list

r= int(input())
l=[]
for i in range(r):
    l.append(list(map(int, input().split())))
min = max = l[0][0]
for row in l:
    for el in row:
        if el < min:
            min = el
        elif el > max:
            max = el
print(min, max)

o/p: 4
6 5 8 9 2
7 6 4
8 12 15 19
9 2
2 6 9 2 19

* generate a 5-digit pin by using given (nx4) matrix first 4-digits in a pin is maximum values from every column and 5th digit is minimum value from whole values.

n= int(input())
l=[]
for i in range(n):
    l.append(list(map(int, input().split())))

```
pin = 0
min = l[0][0]
for j in range(4):
    max = l[0][j]
    for i in range (n):
        if l[i][j] > max
            max = l[i][j]
        if l[i][j] < min
            min = l[i][j]
    pin = pin * 10 + max
pin = pin * 10 + min
print (pin)
```

```
o/p: 3
     0  1  2  3
  0  5  8  2  1
  1  4  6  9  8
  2  7  5  4  3

78981
```

```
l[0][0] = 5
   [0] = 4
   [2]
```

* Matrix addition

```
r = int (input())
c = int (input())
A = [[int(input()) for j in
      range(c)] for i in range(r)]
B = [[int(input()) for j in range(c)]
                   for i in range(r)]
C = [[A[i][j]+B[i][j] for j in range(c)] for i in range(r)]
print(C)
```

$$A = \begin{matrix} \\ 0 \\ 1 \end{matrix}\begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \end{bmatrix} \quad B = \begin{matrix} 0 \\ 1 \end{matrix}\begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \end{bmatrix}$$

$r \times c$ $\quad r \times c$

$c[0][0] = a_{00} + b_{00}$

$c_{01} = a_{01} + b_{01}$

$c_{02} = a_{02} + b_{02}$

$\vdots$

$c_{ij} = a_{ij} + b_{ij}$

o/p: 2
3

$\begin{pmatrix} 4 \\ 5 \\ 1 \end{pmatrix}$ $\begin{bmatrix} 5 \\ 2 \\ 3 \end{bmatrix}$ $\begin{pmatrix} 8 \\ 9 \\ 2 \end{pmatrix}$ $\begin{bmatrix} 4 \\ 6 \\ 7 \end{bmatrix}$

$[9, 7, 4]$ $[12, 15, 9]$

$\begin{bmatrix} 4 & 5 & 1 \\ 8 & 9 & 2 \end{bmatrix}$ $\begin{bmatrix} 5 & 2 & 3 \\ 4 & 6 & 7 \end{bmatrix}$

Matrix multiplication

```
r1 = int(input())
c1 = int(input())
r2 = int(input())
c2 = int(input())
if c1 == r2:
    A = [[int(input()) for j in range
            (c1)] for i in range(r1)]
    B = [[int(input()) for j in range
            (c2)] for i in range(r2)]
    C = []
    for i in range(r1):
        x = []
        for j in range(c2):
            S = 0
            for k in range(c1):
                S = S + A[i][k] * B[k][j]
            x.append(S)
        C.append(x)
    print(C)
else:
    print("mul not possible")
```

(or)

```
C = [[0] * c2 for i in range(r1)]
for i in range(r1):
    for j in range(c2):
        for k in range(c1):
```

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix} \quad B = \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \end{bmatrix}$$

$3 \times 3 \qquad 3 \times 2$

$r_1 \times c_1 \qquad r_2 \times c_2$

$$C = \begin{bmatrix} c_{00} & c_{01} \\ c_{10} & c_{11} \end{bmatrix}$$

$r_1 \times c_2$

3×2

$c_{00} = a_{00} * b_{00} + a_{01} * b_{10} + a_{02} * b_{20}$

$c_{01} = a_{00} * b_{01} + a_{11} * b_{11} + a_{12} * b_{21}$

$\vdots$

$c_{ij} = a_{i0} * b_{0j} + a_{i1} * b_{1j} + a_{i2} * b_{2j}$

$$c_{ij} = \sum_{k=0}^{c_1/r_2} a_{ik} * b_{kj}$$

$\begin{bmatrix} 1 & 2 \end{bmatrix} \quad \begin{bmatrix} 3 & 4 \end{bmatrix}$

$\begin{bmatrix} 3 & 4 \\ 6 & 8 \end{bmatrix}$

o/p:

```
2
1
1
2
1
2
3
4
[3, 4]
[6, 8]
```

* l = [2, 4.7, [5, 8, 7, 3], 2, [2, 6], 5, 9]

2 + 4.7 + 5 + 8 + 7 + 3 + 2 + 2 + 6 + 5 + 9 = 53.7

```
l = list(map(eval, input().split()))  (or)  l = eval(in
s = 0
for i in range(l):
    if type(i) == list:
        s = s + sum(i)
    else:
        s = s + i
print(s)
```

o/p :- [2, 4.7, [5, 8, 7, 3], 2, [2, 6], 5, 9]

53.7

* Read n student details

```
n = int(input())
l = []
for i in range(n):
    l.append([int(input()), input(), input(),
              float(input())])
print(l)
```

o/p :- 2
101
mani
cse
89.7
102
kappa
ece
89.5

[[101, 'mani', 'cse', 89.7], [102, 'kappa', 'ece', 89.5]]

* By default sort of sorted() will sort based on first column.

print(sorted(l, reverse=True))

* sort student details based on name wise.

print(sorted(l, key = lambda x : x[1]))

* sort student details department wise within the department mark wise.

print(sorted(l, key = lambda x:(x[2], x[3])))

Tuple :-

Tuple is a collection of elements.

Tuple is immutable it is faster than list.

* Accessing data from tuple:

```
t = (5, 8, 2, 4, 76, 9)
print(t, type(t))
for i in range(len(t)):
    print(t[i])
for el in t:
    print(el)
print(*t)
```

* searching data in a tuple

```
t = (5, 7, 3, 5, 8, 6)
print(3 in t)
print(12 in t)
print(t.index(8))
```

o/p: True
False
4

* t = (5, 7, 3, 5, 8, 6)
print (sorted (t))

o/p: [3, 5, 5, 6, 7, 8]

* t = (5, 7, 3, 5, 8, 6)
print (t[:3])
print (t[2:5])
print (t[::-1])

o/p: (5, 7, 3)
(3, 5, 8)
(6, 8, 5, 3, 7, 5)

* t = (5, 7, 3, 5, 8, 6)
print (min(t))
print (max(t))
print (sum(t))
print (t.count(5))

o/p: 3
5
34
2

* creating a tuple with one element
t = (5,)
print(t)

o/p: (5,)

Nested tuples:-

* t = ((5, 8, 6), (5, 8), (2, 8, 9))
print (t[0])
print (t[1][1])

o/p: (5, 8, 6)
8

* t=(5, (3,4,5), "sdfrgd", [5,8,3], 9.3)
t[3][0] = 76
print(t)

O/P:- (5,(3,4,5), 'sdfrgd', [76,8,3], 9.3)

* t=(5,8,7,2,6)
t = t + (12,)
print(t)

O/P:- (5,8,7,2,6,12)

* t=(5,8,7,2,6)
t = t[:2] + (12,) + t[2:]
print(t)

O/P:- (5,8,12,7,2,6)

* t=(5,8,7,2,6)
t = t[:2] + t[3:]
print(t)

O/P:- (5,8,2,6)

Set :-

- Set is a collection of elements.
- Set is mutable, elements of set was immutable.
- Set doesn't allow duplicate elements, it doesn't maintain insertion order.
- It doesn't support indexing or subscript operations.

* Access data from set

s = {4,2,23,43,23,46,21,23}
print(s)

O/P:- {4,21,23,7,43,46}

* for el in s:
    print(el)
  print(*s)

* To add the element into set

```
s={6,8,3,65}
s.add(12)
print(s)
```

O/P: {65,3,6,8,12}

* remove the element

```
s={6,8,3,65}
print(s.pop())  #will remove arbitary element
print(s)
```

O/P: 8
{65,3,6}

* To remove the specific element

```
s={6,8,3,65}
s.remove(3)
print(s)
```

O/P: {8,65,6}

* To know element was available in set or not.

```
s={6,8,3,65}
print(3 in s)
```

O/P: True

* creating a set

```
s=set()
for i in range(5):
    s.add(int(input()))
print(s)
```

O/P: 5
3
8
4
2
{5,8,3,4,2}

* s = set(map(int, input().split()))
print(s)

O/P:- 4 5 2 7 8 2 5 4 3
{4, 2, 5, 7, 8}

* s = {int(input()) for i in range(5)}
print(s)

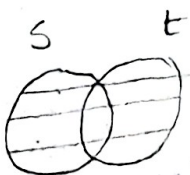
O/P:- 5
4
3
7
8
{3, 4, 5, 7, 8}

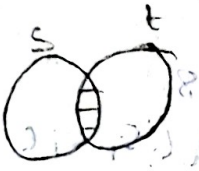
Set operations :-

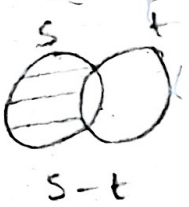
* union

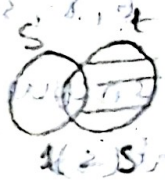
s = {5, 8, 2, 9, 7}
t = {4, 7, 8, 9, 1, 6}
print(s | t)
print(s.union(t))
s.update(t)
print(s)

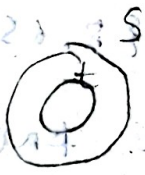
O/P:- {1, 2, 4, 5, 7, 8, 9}

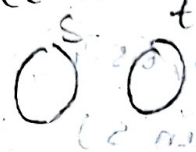












* intersection

s = {5, 8, 2, 9, 7}
t = {4, 7, 8, 9, 1, 6}
print(s & t)
print(s.intersection(t))
s.intersection_update(t)
print(s)

O/P:- {7, 8, 9}

* symmetric difference

```
s = {4,6,3,2,1}
t = {7,6,3,8,9}
print(s^t)
print(s.symmetric_difference(t))
s.symmetric_difference_update(t)
print(s)
```

o/p: {1,2,4,7,8,9}

* set difference

```
s = {4,6,3,2,1}
t = {7,6,3,8,9}
print(s-t)
print(t-s)
print(s.difference(t))
s.difference_update(t)
```

* subset

```
s = {5,7,6,4,3,2}
t = {4,6,5}
print(t<s)
print(t.issubset(s))
```

o/p: true

* Disjoint

```
s = {5,7,6,4,3,2}
t = {1,8,9}
print(s.isdisjoint(t))
```

o/p: true