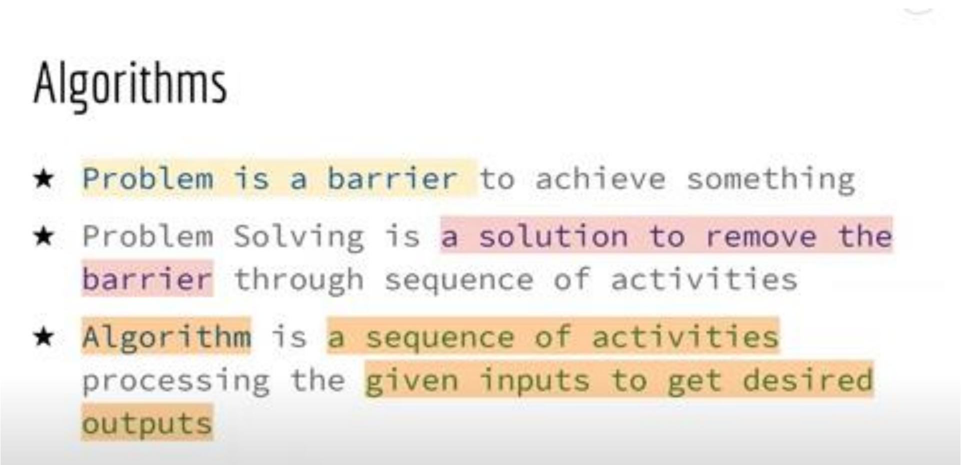
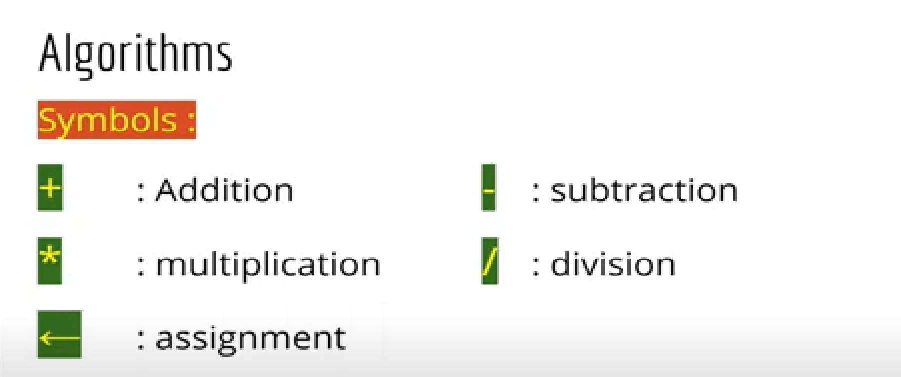


Algorithms

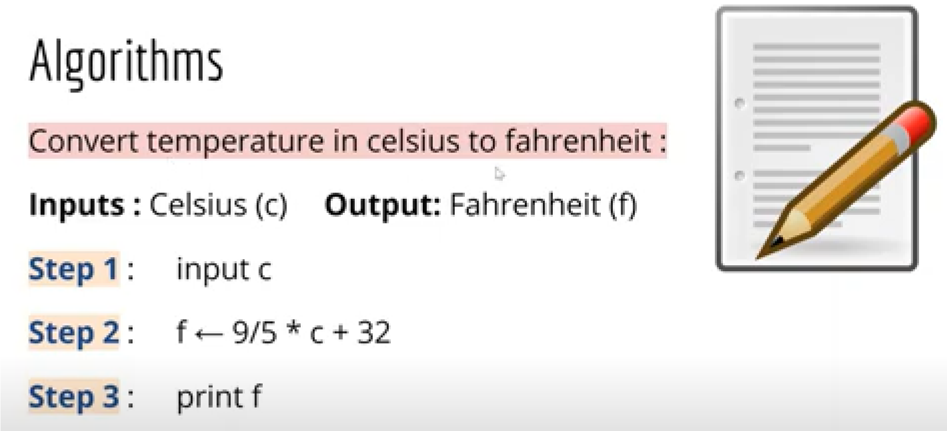
- ★ Problem is a barrier to achieve something
- ★ Problem Solving is a solution to remove the barrier through sequence of activities
- ★ Algorithm is a sequence of activities processing the given inputs to get desired outputs



Algorithms

Symbols :

+	: Addition	-	: subtraction
*	: multiplication	/	: division
←	: assignment		

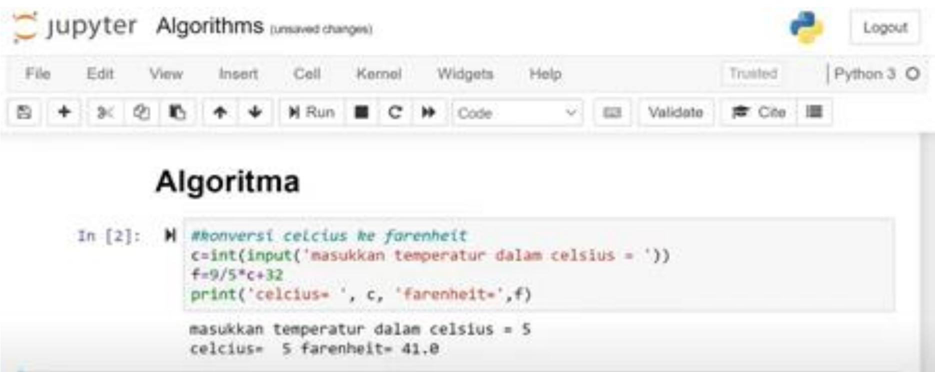


Algorithms

Convert temperature in celsius to fahrenheit :

Inputs : Celsius (c) **Output:** Fahrenheit (f)

```
Step 1 :   input c
Step 2 :   f ← 9/5 * c + 32
Step 3 :   print f
```

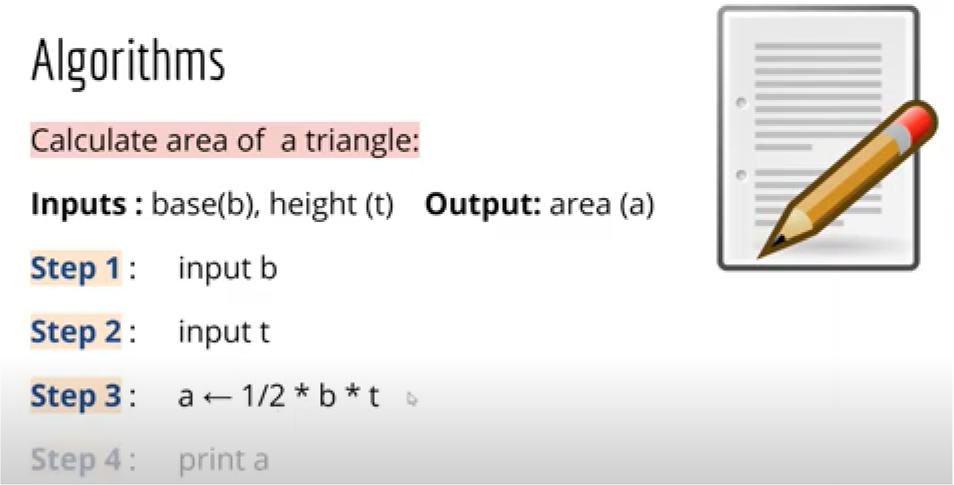


Algoritma

In [2]:

```
#konversi celcius ke farenheit
c=int(input('masukkan temperatur dalam celsius = '))
f=9/5*c+32
print('celcius= ', c, 'farenheit=',f)
```

```
masukkan temperatur dalam celsius = 5
celcius=  5 farenheit= 41.0
```

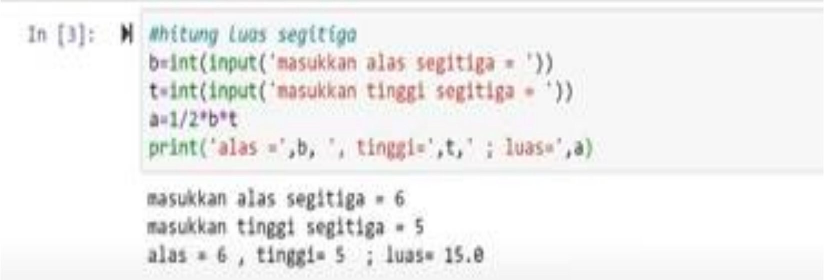


Algorithms

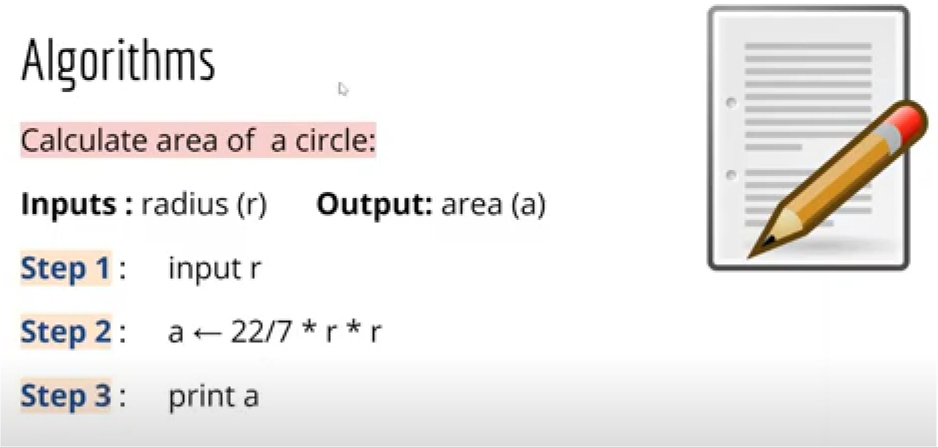
Calculate area of a triangle:

Inputs : base(b), height (t)   **Output:** area (a)

```
Step 1 :    input b
Step 2 :    input t
Step 3 :    a ← 1/2 * b * t
Step 4 :    print a
```



```
In [3]:  #hitung luas segitiga
         b=int(input('masukkan alas segitiga = '))
         t=int(input('masukkan tinggi segitiga = '))
         a=1/2*b*t
         print('alas =',b, ', tinggi=',t,' ; luas=',a)

         masukkan alas segitiga = 6
         masukkan tinggi segitiga = 5
         alas = 6 , tinggi= 5  ; luas= 15.0
```

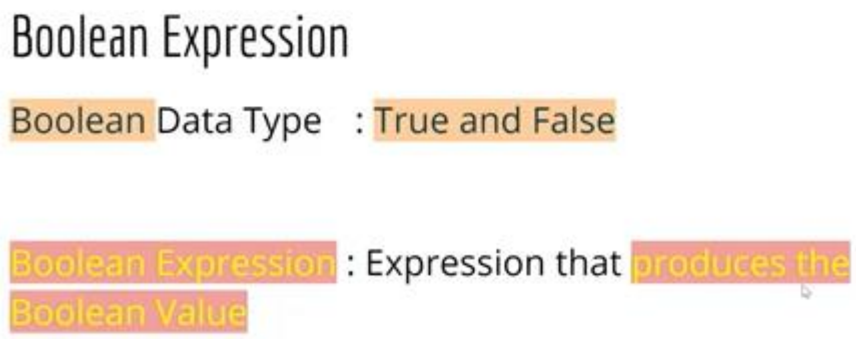


Algorithms

Calculate area of a circle:

Inputs : radius (r)   **Output:** area (a)

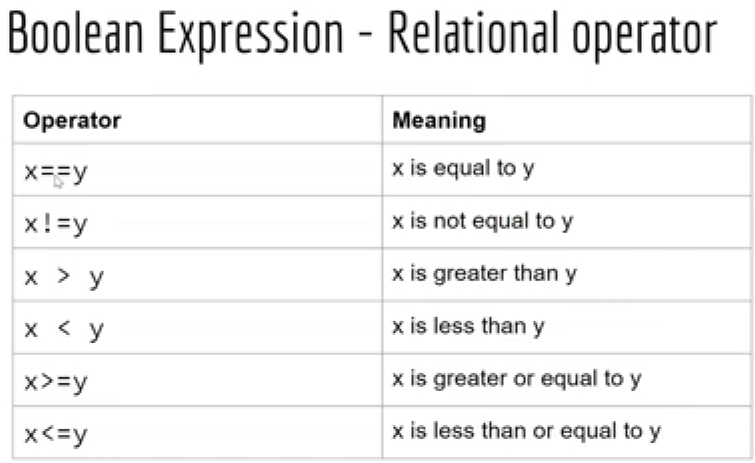
```
Step 1 :    input r
Step 2 :    a ← 22/7 * r * r
Step 3 :    print a
```



Boolean Expression

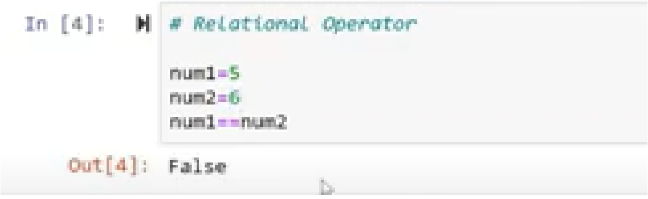
Boolean Data Type   : True and False

Boolean Expression : Expression that produces the Boolean Value

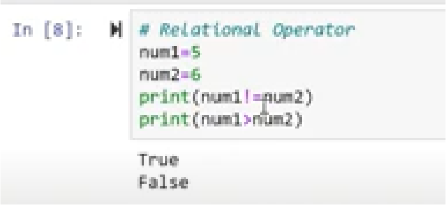


Boolean Expression - Relational operator

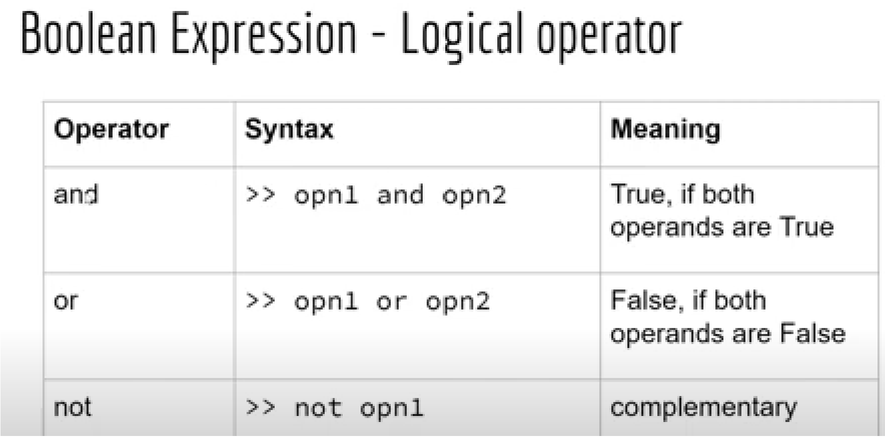
Operator	Meaning
`x==y`	x is equal to y
`x!=y`	x is not equal to y
`x > y`	x is greater than y
`x < y`	x is less than y
`x>=y`	x is greater or equal to y
`x<=y`	x is less than or equal to y



```
In [4]:    # Relational Operator

           num1=5
           num2=6
           num1==num2

   Out[4]: False
```

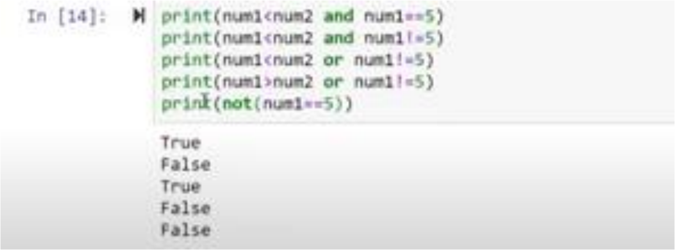


```
In [8]:    # Relational Operator
           num1=5
           num2=6
           print(num1!=num2)
           print(num1>num2)

           True
           False
```



Boolean Expression - Logical operator

Operator	Syntax	Meaning
and	`>> opn1 and opn2`	True, if both operands are True
or	`>> opn1 or opn2`	False, if both operands are False
not	`>> not opn1`	complementary

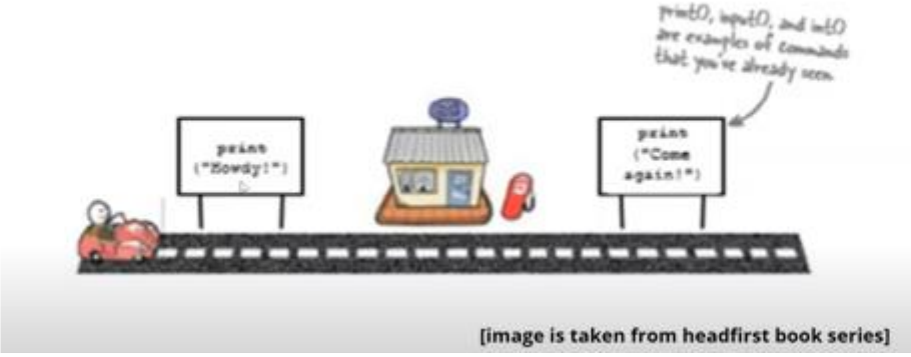


```
In [14]:   print(num1<num2 and num1==5)
           print(num1<num2 and num1!=5)
           print(num1<num2 or num1!=5)
           print(num1>num2 or num1!=5)
           print(not(num1==5))

           True
           False
           True
           False
           False
```

Type of Algorithms

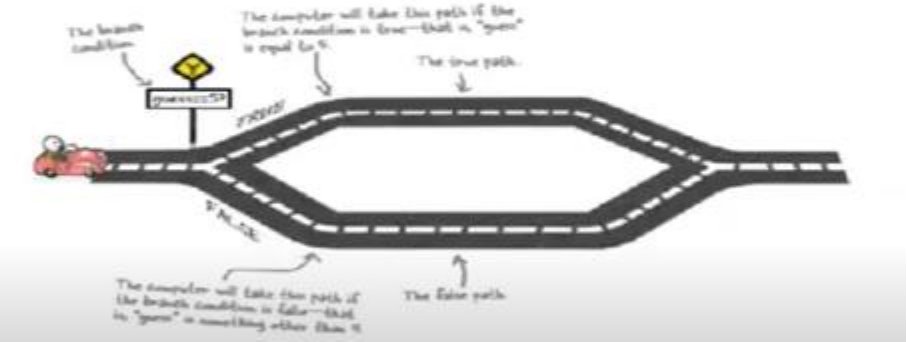
1. Sequence
2. Branching (selection)
3. Loop (Repetition)

Sequence


print("Howdy!")
print("Come again!")
print(), input(), and int() are examples of commands that you've already seen.

[image is taken from headfirst book series]

Branching (Selection)

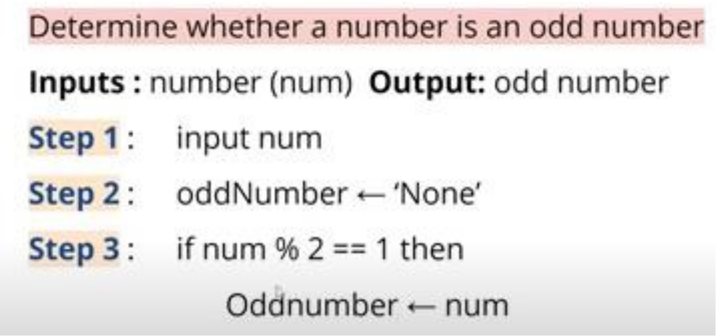

The branch condition
guess==5?
The computer will take this path if the branch condition is true—that is, "guess" is equal to 5.
The true path.
TRUE
FALSE
The computer will take this path if the branch condition is false—that is, "guess" is something other than 5
The false path

Branching (Selection)

Branch Condition : Boolean Expression

```
If BranchCondition is True :
   Syntax_1
   ...
   Syntax_n
```

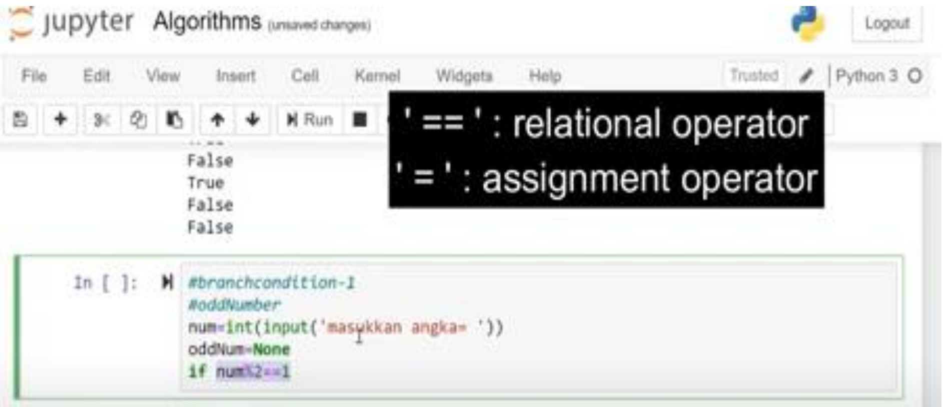
Branching (Selection)



Determine whether a number is an odd number

Inputs : number (num) **Output:** odd number

```
Step 1 :   input num
Step 2 :   oddNumber ← 'None'
Step 3 :   if num % 2 == 1 then
               Oddnumber ← num
```



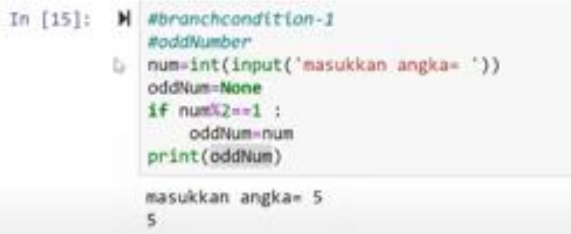
jupyter Algorithms (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run

' == ' : relational operator

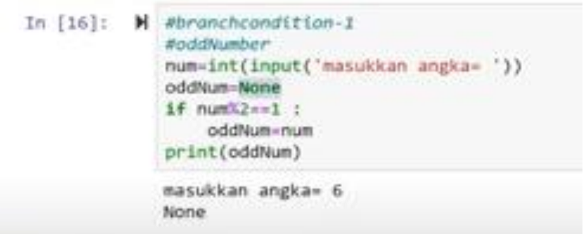
' = ' : assignment operator

```
False
True
False
False
```

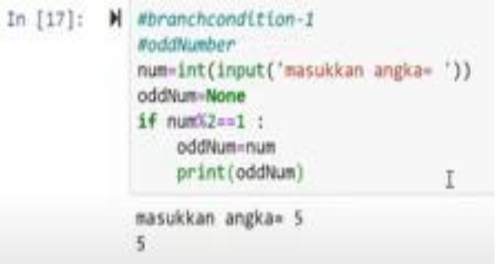
```
In [ ]: #branchcondition-1
        #oddNumber
        num=int(input('masukkan angka= '))
        oddNum=None
        if num%2==1
```



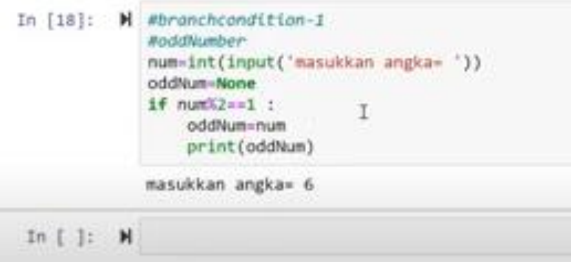
```
In [15]: #branchcondition-1
         #oddNumber
         num=int(input('masukkan angka= '))
         oddNum=None
         if num%2==1 :
             oddNum=num
         print(oddNum)

         masukkan angka= 5
         5
```



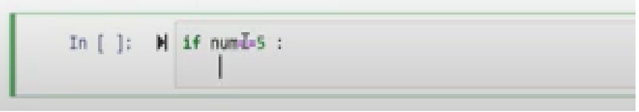
```
In [16]: #branchcondition-1
         #oddNumber
         num=int(input('masukkan angka= '))
         oddNum=None
         if num%2==1 :
             oddNum=num
         print(oddNum)

         masukkan angka= 6
         None
```



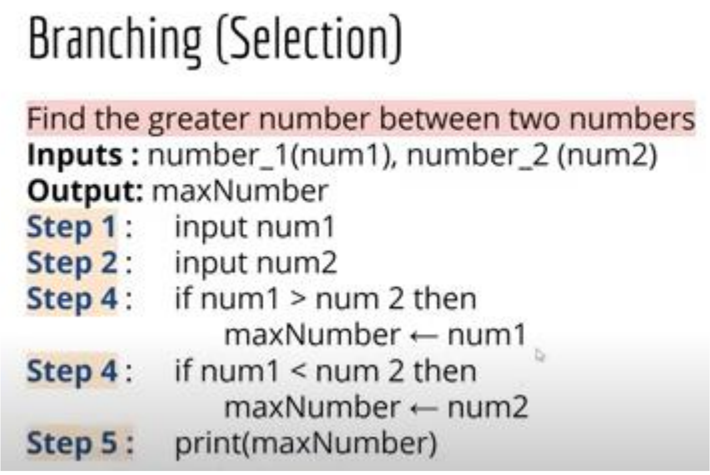
```
In [17]: #branchcondition-1
         #oddNumber
         num=int(input('masukkan angka= '))
         oddNum=None
         if num%2==1 :
             oddNum=num
             print(oddNum)

         masukkan angka= 5
         5
```



```
In [18]: #branchcondition-1
         #oddNumber
         num=int(input('masukkan angka= '))
         oddNum=None
         if num%2==1 :
             oddNum=num
             print(oddNum)

         masukkan angka= 6

In [ ]:
```



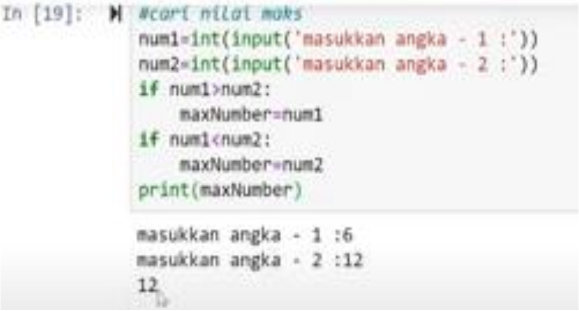
```
In [ ]: if num==5 :
```



Branching (Selection)

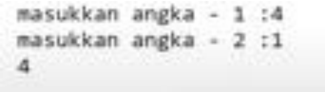
Find the greater number between two numbers
Inputs : number_1(num1), number_2 (num2)
Output: maxNumber

```
Step 1 :   input num1
Step 2 :   input num2
Step 4 :   if num1 > num 2 then
               maxNumber ← num1
Step 4 :   if num1 < num 2 then
               maxNumber ← num2
Step 5 :   print(maxNumber)
```

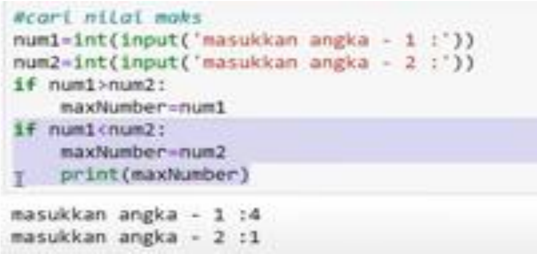


```
In [19]:  #cari nilai maks
          num1=int(input('masukkan angka - 1 :'))
          num2=int(input('masukkan angka - 2 :'))
          if num1>num2:
              maxNumber=num1
          if num1<num2:
              maxNumber=num2
          print(maxNumber)
```

```
masukkan angka - 1 :6
masukkan angka - 2 :12
12
```

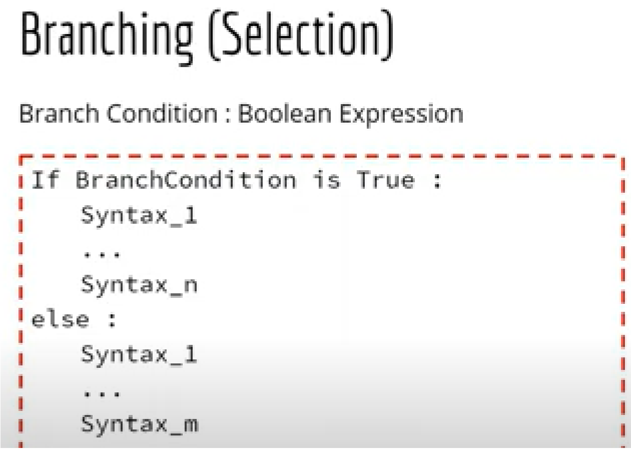


```
masukkan angka - 1 :4
masukkan angka - 2 :1
4
```



```
#cari nilai maks
num1=int(input('masukkan angka - 1 :'))
num2=int(input('masukkan angka - 2 :'))
if num1>num2:
    maxNumber=num1
if num1<num2:
    maxNumber=num2
    print(maxNumber)
```

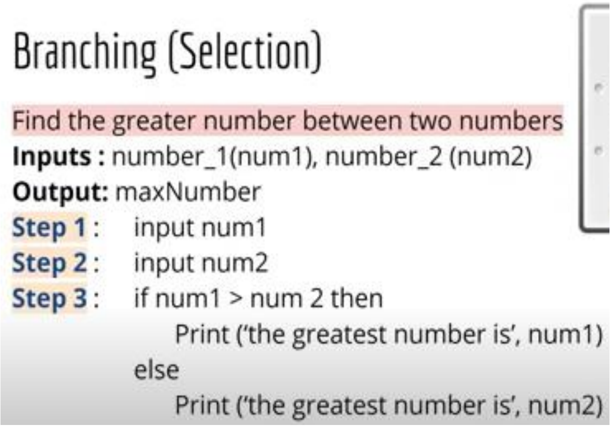
```
masukkan angka - 1 :4
masukkan angka - 2 :1
```



Branching (Selection)

Branch Condition : Boolean Expression

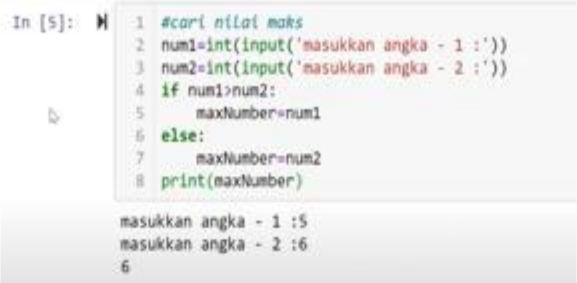
```
If BranchCondition is True :
    Syntax_1
    ...
    Syntax_n
else :
    Syntax_1
    ...
    Syntax_m
```



Branching (Selection)

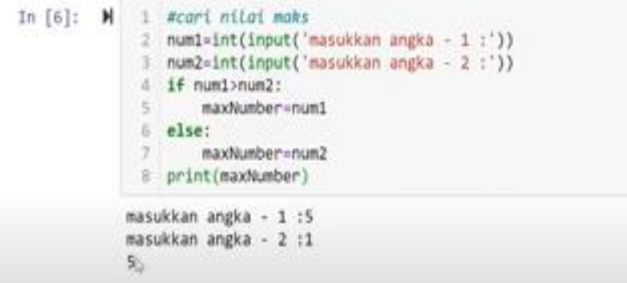
Find the greater number between two numbers
Inputs : number_1(num1), number_2 (num2)
Output: maxNumber

```
Step 1 :   input num1
Step 2 :   input num2
Step 3 :   if num1 > num 2 then
               Print ('the greatest number is', num1)
           else
               Print ('the greatest number is', num2)
```



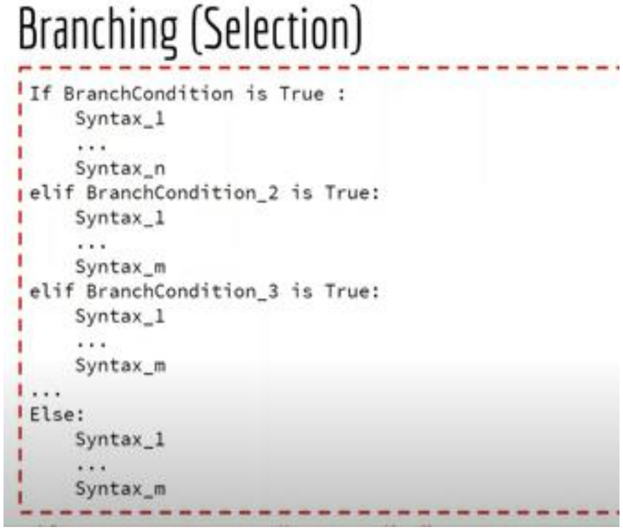
```
In [5]:
#cari nilai maks
num1=int(input('masukkan angka - 1 :'))
num2=int(input('masukkan angka - 2 :'))
if num1>num2:
    maxNumber=num1
else:
    maxNumber=num2
print(maxNumber)
```

```
masukkan angka - 1 :5
masukkan angka - 2 :6
6
```



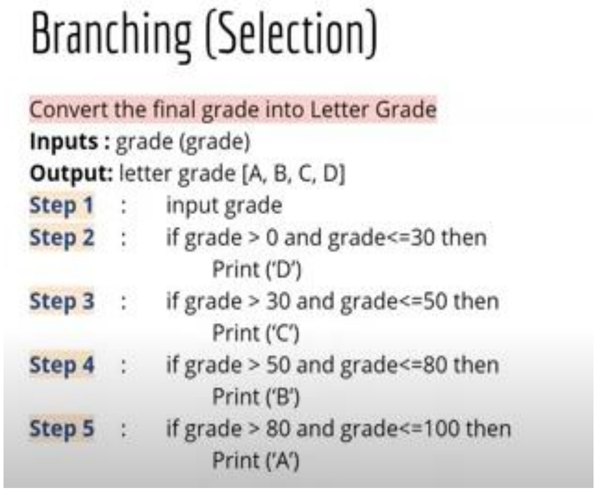
```
In [6]:
#cari nilai maks
num1=int(input('masukkan angka - 1 :'))
num2=int(input('masukkan angka - 2 :'))
if num1>num2:
    maxNumber=num1
else:
    maxNumber=num2
print(maxNumber)
```

```
masukkan angka - 1 :5
masukkan angka - 2 :1
5
```



Branching (Selection)

```
If BranchCondition is True :
    Syntax_1
    ...
    Syntax_n
elif BranchCondition_2 is True:
    Syntax_1
    ...
    Syntax_m
elif BranchCondition_3 is True:
    Syntax_1
    ...
    Syntax_m
...
Else:
    Syntax_1
    ...
    Syntax_m
```



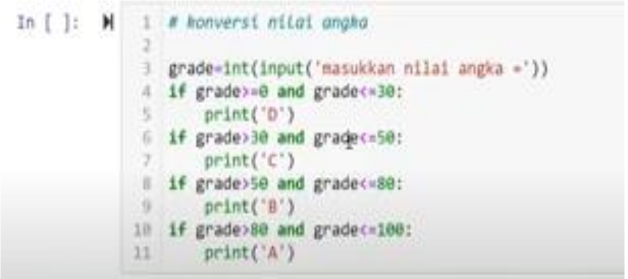
Branching (Selection)

Convert the final grade into Letter Grade

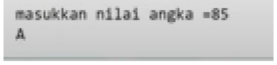
Inputs : grade (grade)

Output: letter grade [A, B, C, D]

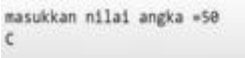
```
Step 1  :  input grade
Step 2  :  if grade > 0 and grade<=30 then
               Print ('D')
Step 3  :  if grade > 30 and grade<=50 then
               Print ('C')
Step 4  :  if grade > 50 and grade<=80 then
               Print ('B')
Step 5  :  if grade > 80 and grade<=100 then
               Print ('A')
```



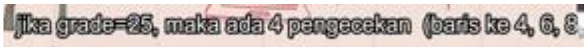
```
In [ ]:
1   # konversi nilai angka
2
3   grade=int(input('masukkan nilai angka ='))
4   if grade>=0 and grade<=30:
5       print('D')
6   if grade>30 and grade<=50:
7       print('C')
8   if grade>50 and grade<=80:
9       print('B')
10  if grade>80 and grade<=100:
11      print('A')
```



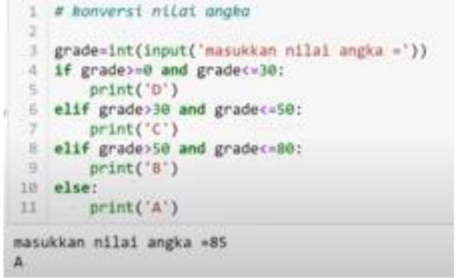
```
masukkan nilai angka =85
A
```



```
masukkan nilai angka =50
C
```

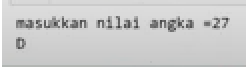


jika grade=25, maka ada 4 pengecekan (baris ke 4, 6, 8 10



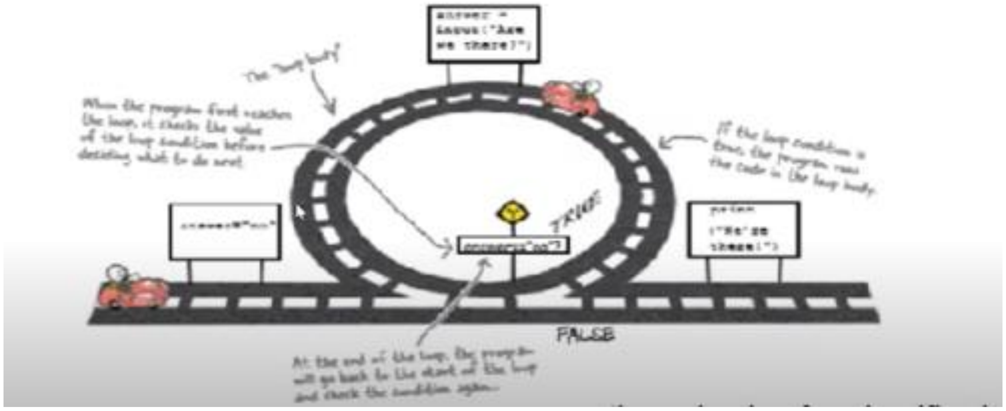
```
1   # konversi nilai angka
2
3   grade=int(input('masukkan nilai angka ='))
4   if grade>=0 and grade<=30:
5       print('D')
6   elif grade>30 and grade<=50:
7       print('C')
8   elif grade>50 and grade<=80:
9       print('B')
10  else:
11      print('A')
```

```
masukkan nilai angka =85
A
```



```
masukkan nilai angka =27
D
```

Loop, Iteration, Repetition

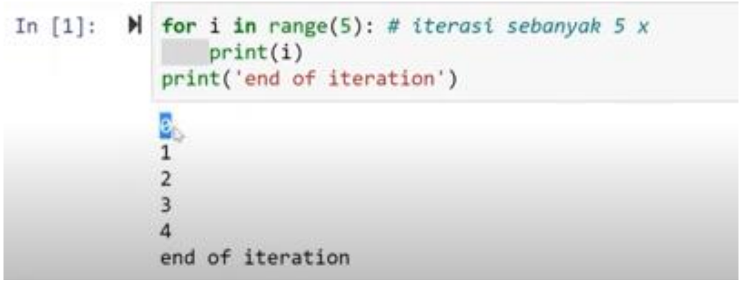

answer = input("Are we there?")
The loop body
When the program first reaches the loop, it checks the value of the loop condition before deciding what to do next
answer="no"
answer=="no"?
TRUE
If the loop condition is true, the program runs the code in the loop body
print ("We're there!")
FALSE
At the end of the loop, the program will go back to the start of the loop and check the condition again

Loop, Iteration, Repetition

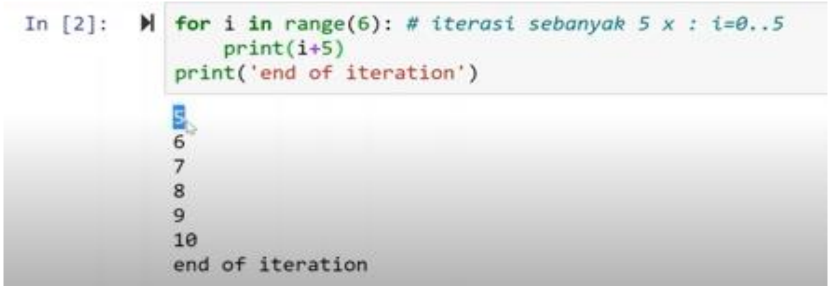
Number of Iteration : known

```
for iterationVar in range (numOfIteration):
    Syntax_1
    Syntax_2
    …
    Syntax_n
```

iterationVar will iterate from 0 (zero) to numOfIteratio-1

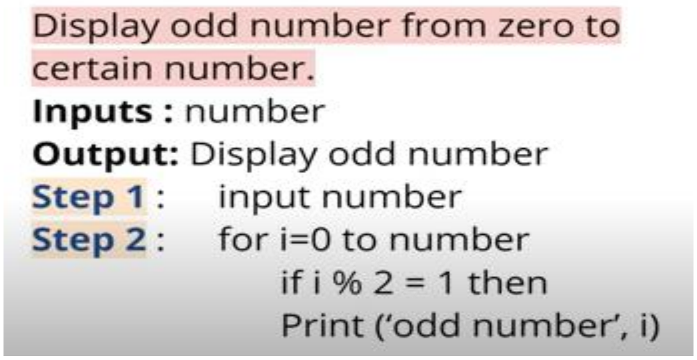


```
In [1]:    for i in range(5): # iterasi sebanyak 5 x
               print(i)
           print('end of iteration')

           0
           1
           2
           3
           4
           end of iteration
```



```
In [2]:    for i in range(6): # iterasi sebanyak 5 x : i=0..5
               print(i+5)
           print('end of iteration')

           5
           6
           7
           8
           9
           10
           end of iteration
```

Loop, Iteration, Repetition

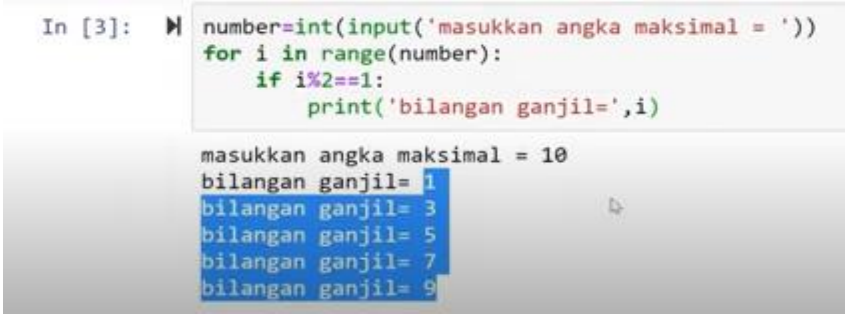


Display odd number from zero to certain number.

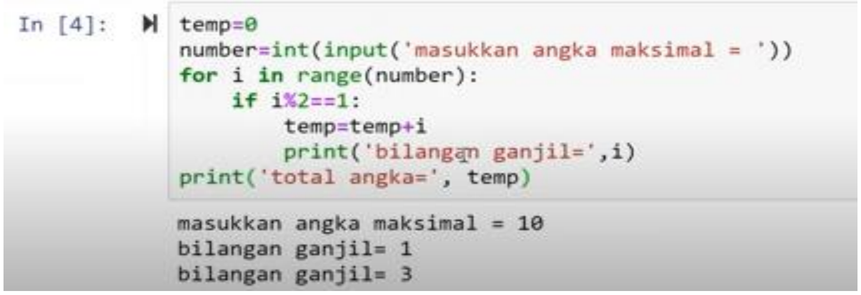
Inputs : number

Output: Display odd number

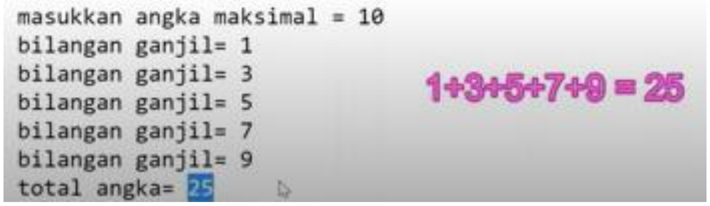
```
Step 1 :    input number
Step 2 :    for i=0 to number
                if i % 2 = 1 then
                Print ('odd number', i)
```



```
In [3]:  number=int(input('masukkan angka maksimal = '))
         for i in range(number):
             if i%2==1:
                 print('bilangan ganjil=',i)

         masukkan angka maksimal = 10
         bilangan ganjil= 1
         bilangan ganjil= 3
         bilangan ganjil= 5
         bilangan ganjil= 7
         bilangan ganjil= 9
```

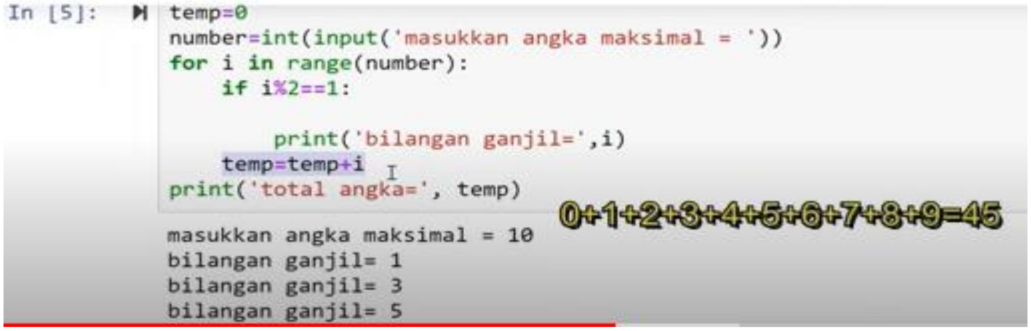


```
In [4]:  temp=0
         number=int(input('masukkan angka maksimal = '))
         for i in range(number):
             if i%2==1:
                 temp=temp+i
                 print('bilangan ganjil=',i)
         print('total angka=', temp)

         masukkan angka maksimal = 10
         bilangan ganjil= 1
         bilangan ganjil= 3
```



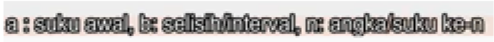
```
masukkan angka maksimal = 10
bilangan ganjil= 1
bilangan ganjil= 3
bilangan ganjil= 5
bilangan ganjil= 7
bilangan ganjil= 9
total angka= 25
```

1+3+5+7+9 = 25

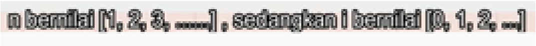


```
In [5]:  temp=0
         number=int(input('masukkan angka maksimal = '))
         for i in range(number):
             if i%2==1:

                 print('bilangan ganjil=',i)
             temp=temp+i
         print('total angka=', temp)

         masukkan angka maksimal = 10
         bilangan ganjil= 1
         bilangan ganjil= 3
         bilangan ganjil= 5
```

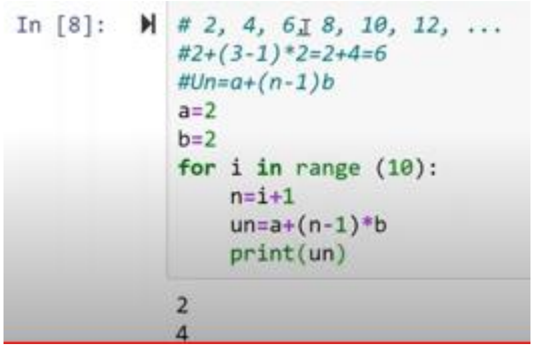
0+1+2+3+4+5+6+7+8+9=45



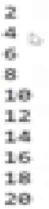
a : suku awal, b: selisih/interval, n: angka/suku ke-n



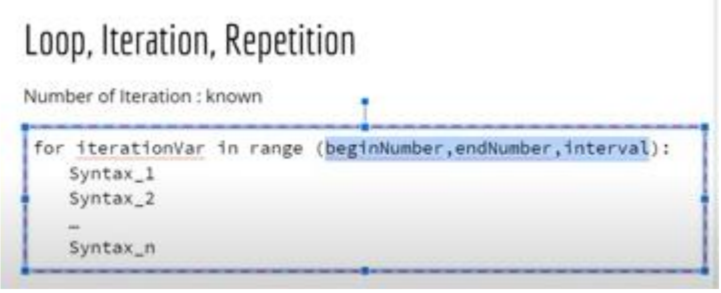
n bernilai [1, 2, 3, ...] , sedangkan i bernilai [0, 1, 2, ...]



```
In [8]:  # 2, 4, 6, 8, 10, 12, ...
         #2+(3-1)*2=2+4=6
         #Un=a+(n-1)b
         a=2
         b=2
         for i in range (10):
             n=i+1
             un=a+(n-1)*b
             print(un)

         2
         4
```



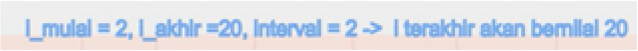
```
2
4
6
8
10
12
14
16
18
20
```



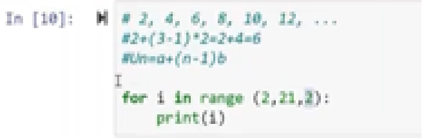
Loop, Iteration, Repetition

Number of Iteration : known

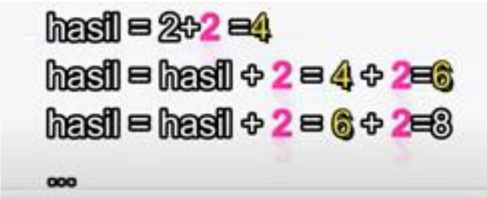
```
for iterationVar in range (beginNumber,endNumber,interval):
    Syntax_1
    Syntax_2
    …
    Syntax_n
```



I_mulai = 2, I_akhir =20, Interval = 2 -> I terakhir akan bernilai 20



```
In [10]:  # 2, 4, 6, 8, 10, 12, ...
          #2+(3-1)*2=2+4=6
          #Un=a+(n-1)b

          for i in range (2,21,2):
              print(i)
```

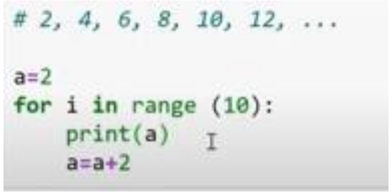


hasil = 2+2 =4

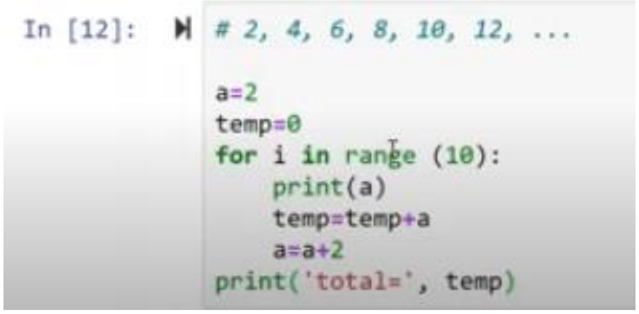
hasil = hasil + 2 = 4 + 2=6

hasil = hasil + 2 = 6 + 2=8

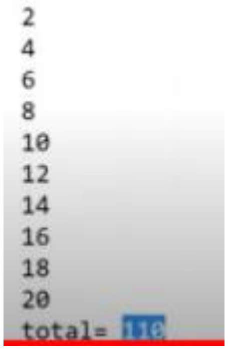
ooo



```
# 2, 4, 6, 8, 10, 12, ...

a=2
for i in range (10):
    print(a)
    a=a+2
```



```
In [12]:  # 2, 4, 6, 8, 10, 12, ...

          a=2
          temp=0
          for i in range (10):
              print(a)
              temp=temp+a
              a=a+2
          print('total=', temp)
```



```
2
4
6
8
10
12
14
16
18
20
total= 110
```