In [ ]:
```
python:it is a high level programming which is known for its simplicity and read
it is an open source environment
python programming was introduced by gudio van rossum in the year 1991
```

In [ ]:
```
use cases of python:
data science/ml-pandas,scikit-learn,tensorflow
website development
app development
game development
automation
cloud and devops
cyber security
iot(internet of things)
sofware development
generative ai
scientific computing
```

In [ ]:
```
Advantages of python:
1.easy to learn
2.open source
3.easy to debug
4.faster development
5.cross platform
6.portable
7.security
8.platform independent
9.libraries
10.community support
11.simple syntax
12.rapid development
13.modern technologies
14.easy to use
15.support multiple languages
16.versatality
17.integration capabilities
```

In [ ]:
```
rules of programming:
1.each python programming should start with column position 1
```

In [1]:
```
x="python"
   y="hello"
```

```
  Cell In[1], line 2
    y="hello"
    ^
IndentationError: unexpected indent
```

In [ ]:
```
2.variables are case sensitive
```

In [3]:
```
def="my world"
print(def)
```

```
  Cell In[3], line 1
    def="my world"
        ^
SyntaxError: invalid syntax
```

In [5]:
```
help('keywords')
```

```
Here is a list of the Python keywords.  Enter any keyword to get more help.

False               class               from                or
None                continue            global              pass
True                def                 if                  raise
and                 del                 import              return
as                  elif                in                  try
assert              else                is                  while
async               except              lambda              with
await               finally             nonlocal            yield
break               for                 not
```

In [ ]:
```
3.if our statement is expecting some sub statement then the line should end with
```

In [7]:
```
x=10
if x>6:
print("welcome")
```

```
  Cell In[7], line 2
    if x>6
          ^
SyntaxError: expected ':'
```

In [ ]:
```
4.every substement should start at forwarded position to its parent line
```

In [9]:
```
x=10
if x>6:
print("welcome")
```

```
  Cell In[9], line 3
    print("welcome")
    ^
IndentationError: expected an indented block after 'if' statement on line 2
```

In [ ]:
```
5.all the sub statements of a parent line should be in the same position
```

In [11]:
```
a=10
b=20
if a<b:
    print("hi")
elif a>b:
    print("hello")
else:
    print("nor hi or nor hello")
```

```
hi
```

In [ ]:
```
variables:it is used to store the data(any thing)
rules of variables:
cannot start with number
spaces are not allowed
special characters are not allowed
its a case-sensitive
keywords or reserve words cannot used as a variable
must start with letter/_
```

In [13]:
```
myvar1="hello"
print(myvar1)
```

```
hello
```

In [15]:
```
$myvar=10
print($myvar)
```

```
  Cell In[15], line 1
    $myvar=10
    ^
SyntaxError: invalid syntax
```

In [17]:
```
@char="python"
print(@char)
```

```
  Cell In[17], line 1
    @char="python"
     ^
SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?
```

In [19]:
```
except="hello world"
print(except)
```

```
  Cell In[19], line 1
    except="hello world"
    ^
SyntaxError: invalid syntax
```

In [ ]:
```
data types:
it is used to store the type of the data in a variable

data types are classified into different types:

1.numeric types:int,float,complex
2.sequence types:list,tuple,range
3.set types:set,frozenset
4.mapping type:dict
5.binary types:byte,bytearray,memoryview
6.bool types:True,False
7.text type:string
```

In [21]:
```
#numeric types:
x=89
y=10.5
z=2j
print(x,type(x))
print(y,type(y))
print(z,type(z))
```

```
89 <class 'int'>
10.5 <class 'float'>
2j <class 'complex'>
```

In [23]:
```
cars=["volvo","bmw","ford","ertiga"]
print(cars,type(cars))
```

```
['volvo', 'bmw', 'ford', 'ertiga'] <class 'list'>
```

In [25]:
```
cars[2]
```

Out[25]:
```
'ford'
```

In [ ]:
```
slicing:which is used to access the portion of data
start:stop:step(-1)
```

In [27]:
```
cars[:2]
```

Out[27]:
```
['volvo', 'bmw']
```

In [29]:
```
cars[:]
```

Out[29]:
```
['volvo', 'bmw', 'ford', 'ertiga']
```

In [31]:
```
x=(1,5,7,9)
print(x,type(x))
```

```
(1, 5, 7, 9) <class 'tuple'>
```

In [33]:
```
x[0]
```

Out[33]:
```
1
```

In [35]:
```
x[2]
```

Out[35]:
```
7
```

In [ ]:
```
#list methods:
append
pop
remove
insert
extend
reverse
clear
sort
```

In [37]:
```
cars.append("fortuner")
```

In [39]:
```
cars
```

Out[39]:
```
['volvo', 'bmw', 'ford', 'ertiga', 'fortuner']
```

In [41]:
```
x=range(9)
```

In [43]:
```
x
```

Out[43]:
```
range(0, 9)
```

In [45]:
```
for y in x:
    print(y)
```

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

In [51]:
```
x={1,2,3,3,4,5,4,5}
print(x)
print(type(x))
```

```
{1, 2, 3, 4, 5}
<class 'set'>
```

In [53]:
```
x=frozenset({1,2,3,3,4,5,4,5})
print(x)
print(type(x))
```

```
frozenset({1, 2, 3, 4, 5})
<class 'frozenset'>
```

In [55]:
```
y={"Name":"punith","Age":25}
```

In [57]:
```
y
```

Out[57]:
```
{'Name': 'punith', 'Age': 25}
```

In [59]:
```
type(y)
```

Out[59]:
```
dict
```

In [61]:
```
y.keys()
```

Out[61]:
```
dict_keys(['Name', 'Age'])
```

In [63]:
```
y.values()
```

Out[63]:
```
dict_values(['punith', 25])
```

In [65]:
```
str1="welcome to the session"
print(str1)
```

```
welcome to the session
```

In [67]:
```
print(type(str1))
```

```
<class 'str'>
```

In [69]:
```
str1.upper()
```

Out[69]:
```
'WELCOME TO THE SESSION'
```

In [71]:
```
str1.lower()
```

Out[71]:
```
'welcome to the session'
```

In [73]:
```
str1.replace("session","class")
```

Out[73]:
```
'welcome to the class'
```

In [75]:
```
str1.split()
```

Out[75]:
```
['welcome', 'to', 'the', 'session']
```

In [77]:
```
x=True
print(x,type(x))
```

```
True <class 'bool'>
```

In [ ]:
```
task:
Clean raw CSV logs
```

In [ ]:
```
upcoming:
Control Flow
conditions statements
loopings
```