### LANGUAGE

#### FUNDAMENTALS

##### Introduction

* **Pythonisageneralpurposehighlevelprogramminglanguage.**
* **PythonwasdevelopedbyGuidoVanRossamin1989whileworkingatNational Research Institute at Netherlands.**
* **ButofficiallyPythonwasmadeavailabletopublicin1991.TheofficialDateofBirthfor Python is : Feb 20th 1991.**
* **Pythonisrecommendedasfirstprogramminglanguagefor beginners.**

Eg1:ToprintHelloworld Java:

1)publicclassHelloWorld

**6) }**

**4) {**

**2){**

3) ps vmain(String[]args)

5) SOP("Helloworld");

7)}

C:

1)#include<stdio.h>

**4) print("Helloworld");**

**2)voidmain()**

**3){**

**5)}**

Python:

print("HelloWorld")

Eg2:Toprintthesumof2numbers Java:

1)publicclassAdd

**4) {**

**2){**

3) publicstaticvoidmain(String[]args)

**5) inta,b;**

|  |  |  |
| --- | --- | --- |
| **6)** |  | **a =10;** |
| **7)** |  | **b=20;** |
| **8) System.out.println("TheSum:"+(a+b));** | | |
| **9)** | **}** | |
| **10)}** | | |

C:

**1)#include <stdio.h>**

**3)voidmain()**

**5) inta,b;**

**7) b=20;**

**9)}**

**8) printf("TheSum:%d",(a+b));**

**6) a=10;**

**4){**

**2)**

Python:

1)a=10

**2)b=20**

3)print("The Sum:",(a+b))

Thename Pythonwasselected fromtheTVShow

"TheCompleteMontyPython'sCircus",whichwasbroadcastedinBBCfrom1969to 1974.

GuidodevelopedPythonlanguagebytakingalmostallprogrammingfeaturesfrom different languages

1. **FunctionalProgrammingFeaturesfromC**
2. **ObjectOrientedProgrammingFeaturesfromC++**
3. **ScriptingLanguageFeaturesfromPerlandShell Script**
4. **ModularProgrammingFeaturesfromModula-3**

Mostofsyntaxin Python DerivedfromCandABC languages.

WherewecanusePython:

Wecanuseeverywhere.Themostcommon importantapplicationareasare

1. **FordevelopingDesktop Applications**
2. **Fordevelopingweb Applications**
3. **Fordevelopingdatabase Applications**
4. **ForNetworkProgramming**

1. **Fordeveloping games**
2. **For DataAnalysis Applications**
3. **ForMachine Learning**
4. **FordevelopingArtificialIntelligenceApplications**
5. **ForIoT**

**...**

**Note:**

* + **InternallyGoogleandYoutubeusePythoncoding.**
  + **NASAandNeworkStockExchange Applicationsdevelopedby Python.**
  + **TopSoftwarecompanieslikeGoogle,Microsoft,IBM,YahoousingPython.**

Featuresof Python:

1. **Simpleandeasytolearn:**
   * **Pythonisasimpleprogramminglanguage.Whenweread Pythonprogram,wecanfeel like reading english statements.**
   * **Thesyntaxesareverysimpleandonly30+kerywordsareavailable.**
   * **Whencomparedwithother languages,wecanwriteprogramswithverylessnumber of lines. Hence more readability and simplicity.**
   * **Wecanreducedevelopmentandcostofthe project.**
2. **FreewareandOpenSource:**
   * **WecanusePythonsoftwarewithoutanylicenceanditisfreeware.**
   * **Itssourcecodeisopen,so thatwecanwecancustomizebasedonourrequirement.**
   * **Eg:JythoniscustomizedversionofPythontoworkwithJava Applications.**
3. **HighLevelProgramminglanguage:**
   * **Beingaprogrammerwearenotrequiredtoconcentratelowlevelactivitieslike memory management and security etc.**
   * **Pythonishighlevelprogramminglanguageandhenceitisprogrammerfriendly language.**
4. **PlatformIndependent:**
   * **OncewewriteaPythonprogram,itcanrunonanyplatformwithoutrewritingonce again.**
   * **InternallyPVMisresponsibletoconvertintomachineunderstandableform.**
5. **Portability:**

Python programs are portable. ie we can migrate from one platform to another platformveryeasily. Pythonprogramswillprovidesameresultsonanypaltform.



1. **DynamicallyTyped:**
   * **In Python we are not required to declare type for variables. Whenever we are assigningthevalue,basedonvalue,typewillbeallocatedautomatically.Hence Python is considered as dynamically typed language.**
   * **ButJava,CetcareStaticallyTypedLanguagesb'zwehavetoprovidetypeatthe beginning only.**
   * **Thisdynamictypingnaturewillprovidemoreflexibilitytotheprogrammer.**
2. **BothProcedureOrientedandObjectOriented:**

Python language supports both Procedure oriented (like C, pascal etc) and object oriented(likeC++,Java)features.Hencewecangetbenefitsofbothlikesecurityand reusability etc

1. **Interpreted:**
   * **WearenotrequiredtocompilePythonprogramsexplcitly.InternallyPython interpreter will take care that compilation.**
   * **Ifcompilationfailsinterpreterraisedsyntaxerrors.Oncecompilationsuccessthen PVM (Python Virtual Machine) is responsible to execute.**
2. **Extensible:**
   * **WecanuseotherlanguageprogramsinPython.**
   * **Themainadvantagesofthisapproachare:**
     + **Wecan usealreadyexistinglegacynon-Pythoncode**
     + **Wecanimproveperformanceoftheapplication**
3. **Embedded:**

WecanusePythonprogramsinanyotherlanguageprograms.

i.ewecanembeddPythonprogramsanywhere.

1. **ExtensiveLibrary:**
   * **Python hasarichinbuilt library.**
   * **Beingaprogrammerwecanusethislibrarydirectlyandwearenotresponsibleto implement the functionality. Etc.**

LimitationsofPython:

1. **Performancewisenotuptothemarkbecauseitisinterpretedlanguage.**
2. **NotusingformobileApplications.**



FlavorsofPython:

1. **CPython:**

Itisthestandardflavor ofPython.Itcanbeused toworkwith ClanugageApplications.

1. **JythonORJPython:**

ItisforJavaApplications.ItcanrunonJVM

1. **IronPython:**

Itisfor C#.Netplatform

1. **PyPy:**

ThemainadvantageofPyPyisperformancewillbeimprovedbecauseJITcompileris available inside PVM.

1. **RubyPython**

ForRubyPlatforms

1. **AnacondaPython**

Itisspeciallydesignedforhandlinglargevolumeofdataprocessing.

PythonVersions:

* + **Python1.0VintroducedinJan1994**
  + **Python2.0VintroducedinOctober2000**
  + **Python3.0VintroducedinDecember2008**

Note:Python3won'tprovidebackwardcompatibilityto Python2i.ethereisno guarantee that Python2 programs will run in Python3.

Currentversions

Python 3.11.4



#### IDENTIFIERS

* **ANamein Python Programiscalled Identifier.**
* **ItcanbeClassNameORFunctionNameORModuleNameORVariableName.**
* **a=10**

RulestodefineIdentifiersinPython:

1. **TheonlyallowedcharactersinPythonare**
   * **alphabetsymbols(eitherlowercaseoruppercase)**
   * **digits(0to9)**
   * **underscoresymbol(\_)**

Bymistakeifweareusinganyothersymbollike$thenwewillget syntaxerror.

* + **cash=10√**
  + **ca$h =20**

1. **Identifiershouldnotstartswithdigit**
   * **123total** 
   * **total123√**
2. **Identifiersarecasesensitive. OfcoursePythonlanguageiscasesensitivelanguage.**
   * **total=10**
   * **TOTAL=999**
   * **print(total)#10**
   * **print(TOTAL)#999**



Identifier:

1. **AlphabetSymbols(EitherUppercaseORLowercase)**
2. **IfIdentifierisstartwithUnderscore(\_)thenitindicatesitisprivate.**
3. **IdentifiershouldnotstartwithDigits.**
4. **Identifiersarecase sensitive.**
5. **Wecannotusereservedwordsasidentifiers Eg: def = 10** 
6. **Thereisnolengthlimitfor Pythonidentifiers.Butnotrecommendedtousetoo lengthy identifiers.**
7. **Dollor($)SymbolisnotallowedinPython.**
8. **WhichofthefollowingarevalidPython identifiers?**
   1. **123total** 
   2. **total123√**
   3. **java2share√**
   4. **ca$h**
   5. **\_abc\_abc\_√**
   6. **def**
   7. **if**

**Note:**

1. **Ifidentifier startswith\_symbolthenitindicatesthat itisprivate**
2. **Ifidentifierstartswith(TwoUnderScoreSymbols)indicatingthatstronglyprivate identifier.**
3. **Iftheidentifierstartsandendswithtwounderscoresymbolsthentheidentifieris language defined special name, which is also known as magic methods.**
4. **Eg: add**



RESERVEDWORDS

InPythonsomewordsarereservedtorepresentsomemeaningorfunctionality. Such types of words are called reserved words.

Thereare33reservedwordsavailablein Python.

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

**Note:**

1. **AllReservedwordsinPythoncontainonlyalphabetsymbols.**
2. **Exceptthefollowing3reservedwords,allcontainonlylowercasealphabet symbols.**
   * **True**
   * **False**
   * **None**

Eg:a=true

a=True√

>>>import keyword

>>>keyword.kwlist

['False','None','True','and','as','assert','break','class','continue','def','del','elif','else',

'except','finally','for','from','global','if','import','in','is','lambda','nonlocal','not','or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']



#### DATATYPES

* **DataTyperepresents thetypeofdatapresentinsidea variable.**
* **In Pythonwearenot required tospecifythetypeexplicitly.Based onvalueprovided, thetypewillbeassignedautomatically.Hence PythonisdynamicallyTypedLanguage.**

Pythoncontainsthefollowinginbuiltdata types

1. **Int**
2. **Float**
3. **Complex**
4. **Bool**
5. **Str**
6. **Bytes**
7. **Bytearray**
8. **Range**
9. **List**
10. **Tuple**
11. **Set**
12. **Frozenset**
13. **Dict**
14. **None**

**a**

**ab**

**10**

**a= 10**

**b= 10**

**20**

**a= 10**

**a= 20**

**10**

Note:Pythoncontainsseveralinbuilt functions

1. **type()**

to checkthetypeofvariable

1. **id()**

togetaddressofobject



1. **print()**

toprintthevalue

InPythoneverythingisanObject.

###### intData Type:

Wecanuseintdatatypetorepresentwholenumbers(integralvalues)

**Eg:a= 10**

type(a)#int

**Note:**

* + **InPython2wehavelongdatatypetorepresentverylargeintegralvalues.**
  + **Butin Python3thereisnolongtypeexplicitlyandwecanrepresentlongvaluesalsoby using int type only.**

Wecan representintvaluesin thefollowing ways

1. **Decimal form**
2. **Binary form**
3. **Octalform**
4. **Hexadecimalform**
5. DecimalForm(Base-10):
   * **ItisthedefaultnumbersysteminPython**
   * **Thealloweddigitsare:0to9**
   * **Eg:a=10**
6. BinaryForm(Base-2):
   * **Thealloweddigits are:0 &1**
   * **Literalvalueshouldbeprefixedwith0bor0B**
   * **Eg:a= 0B1111**

a=0B123

a=b111

1. OctalForm(Base-8):
   * **Thealloweddigits are:0 to7**
   * **Literalvalueshouldbeprefixedwith0oor0O.**
   * **Eg:a= 0o123**

a =0o786

1. **HexaDecimalForm(Base-16):**
   * **Thealloweddigitsare:0to9,a-f(bothlowerand uppercasesare allowed)**
   * **Literalvalueshouldbeprefixedwith0xor0X**
   * **Eg:a= 0XFACE**

a=0XBeef a=0XBeer

Note:Beingaprogrammerwecanspecifyliteralvaluesindecimal, binary,octalandhexa decimal forms. But PVM will always provide values only in decimal form.

* + - **a=10**
    - **b=0o10**
    - **c=0X10**
    - **d=0B10**
    - **print(a)10**
    - **print(b)8**
    - **print(c)16**
    - **print(d)2**

BaseConversions

Pythonprovidethefollowingin-builtfunctionsfor baseconversions

* 1. bin():

Wecanusebin()toconvertfromanybaseto binary

* + 1. **>>>bin(15)**

|  |
| --- |
| **2)'0b1111'** |
| **3)>>>bin(0o11)** |
| **4)'0b1001'** |
| **5)>>>bin(0X10)** |
| **6)'0b10000'** |

* 1. oct():

Wecanuseoct()toconvert fromanybasetooctal

* + 1. **>>>oct(10)**

|  |
| --- |
| **2)'0o12'** |
| **3)>>>oct(0B1111)** |
| **4)'0o17'** |
| **5)>>>oct(0X123)** |
| 6)**'0o443'** |

* 1. hex():

Wecanusehex()toconvertfromanybasetohexadecimal

* + 1. **>>>hex(100)**

|  |
| --- |
| **2)'0x64'** |
| **3)>>>hex(0B111111)** |
| **4)'0x3f'** |
| **5)>>>hex(0o12345)** |
| 6)**'0x14e5'** |

###### FloatDataType:

* + **Wecanusefloatdatatypetorepresentfloatingpointvalues(decimalvalues)**

**Eg:f= 1.234**

type(f)float

* + **Wecanalsorepresentfloatingpointvaluesbyusingexponentialform (Scientific Notation)**

Eg:f=1.2e3insteadof'e'wecanuse'E' print(f)1200.0

* + **Themainadvantageofexponentialformiswecanrepresentbigvaluesinless memory.**

**\*\*\*Note:**

Wecanrepresentintvaluesindecimal, binary,octalandhexadecimalforms.Butwe can represent float values only by using decimal form.

**1) >>>f=0B11.01**

**3) f=0B11.01**

**5)SyntaxError:invalidsyntax**

**7)>>>f=0o123.456**

**9)**

**11)SyntaxError:invalid syntax**

**10)>>>f=0X123.456**

**8) SyntaxError:invalidsyntax**

**6)**

**^**

**4)**

**2) File"<stdin>",line1**

###### ComplexDataType:

* + **Acomplexnumberisoftheform**



**a+bj**

**j2=-1**

**j=**√−1

**RealPart ImaginaryPart**

* + **‘a’and‘b’containIntergers ORFloatingPointValues.**

**Eg:3 +5j**

10+5.5j

0.5+0.1j

* + **Intherealpartifweuseintvalue thenwecanspecifythateitherbydecimal, octal, binary or hexa decimal form.**
  + **Butimaginarypartshouldbespecifiedonlybyusingdecimal form.**

1)>>>a=0B11+5j

**4)>>>a=3+0B11j**

**2)>>>a**

3)(3+5j)

5) SyntaxError:invalidsyntax

* + **Evenwecanperform operationsoncomplextype values.**

1)>>>a=10+1.5j

**4)>>>print(c)**

**2)>>>b=20+2.5j**

3)>>>c=a+b

5)(30+4j)

7)<class'complex'>

**6)>>>type(c)**

Note:Complexdatatypehassomeinbuiltattributestoretrievetherealpartand imaginary part

c=10.5+3.6j

c.real10.5

c.imag3.6

WecanusecomplextypegenerallyinscientificApplicationsandelectricalengineering Applications.

###### bool Data Type:

* + **Wecan usethisdatatypetorepresentboolean values.**
  + **Theonlyallowedvaluesforthisdatatypeare:**
  + **Trueand False**
  + **InternallyPythonrepresentsTrueas1andFalseas0 b = True**

type(b)bool

**Eg:**

a =10

b=20

c=a<b

print(c)True

True+True2

True-False1

###### strDataType:

* + **strrepresentsStringdatatype.**
  + **AStringisasequenceofcharactersenclosedwithinsinglequotesordouble quotes.**
  + **s1='durga'**
  + **s1="durga"**

* + **Byusingsinglequotesordoublequoteswecannotrepresentmultilinestring literals.**
  + **s1="durga soft"**
  + **Forthisrequirementweshouldgofortriplesinglequotes(''')ortripledouble quotes(""")**
  + **s1='''durga soft'''**
  + **s1="""durga soft"""**
  + **Wecan alsousetriplequotesto usesinglequoteordoublequoteinour String.**
  + **'''Thisis"character'''' This i " Character '**
  + **Wecanembedonestringinanother string**
  + **'''This"Python classveryhelpful"forjava students'''**

SlicingofStrings:

1. **slicemeansapiece**
2. **[]operatoriscalledsliceoperator,which canbeused toretrievepartsof String.**
3. **InPythonStringsfollowszerobasedindex.**
4. **Theindexcanbeeither+veor-ve.**

5)+veindexmeansforwarddirectionfromLefttoRight

1. **-veindexmeansbackwarddirectionfromRighttoLeft**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | | **-5** | **-4** | **-3** | **-2** | **-1** |
| **d** | **u** | **R** | **g** | **a** |
| **1)>>>s="durga"** | | **0** | **1** | **2** | **3** | **4** |
| **2)** | **>>>s[0]** | | | | | | |
| **3)** | **'d'** | | | | | | |
| **4)** | **>>>s[1]** | | | | | | |
| **5)** | **'u'** | | | | | | |
| **6)** | **>>>s[-1]** | | | | | | |
| **7)** | **'a'** | | | | | | |
| **8)** | **>>>s[40]** | | | | | | |

IndexError:stringindexoutofrange

* 1. **>>>s[1:40]**

|  |
| --- |
| **2)'urga'** |
| **3)>>>s[1:]** |
| **4)'urga'** |
| **5)>>>s[:4]** |
| **6)'durg'** |
| **7)>>>s[:]** |
| 1. **'durga'** 2. **>>>** |
| **10)** |
| **11)>>>s\*3** |
| **12)'durgadurgadurga'** |
| **13)** |
| **14)>>>len(s)** |
| **15)5** |

**Note:**

1. **InPythonthefollowingdatatypesareconsideredasFundamentalData types**
   * **int**
   * **float**
   * **complex**
   * **bool**
   * **str**
2. **InPython,wecanrepresentcharvaluesalsobyusingstrtypeand explicitlychartype is not available.**

1)>>>c='a'

**2)>>>type(c)**

3)<class'str'>

1. **longDataTypeisavailablein Python2butnotinPython3.InPython3longvaluesalso we can represent by using int type only.**
2. **InPythonwecanpresentcharValuealsobyusingstrTypeandexplicitlycharTypeis not available.**



#### TYPECASTING

֍Wecanconvertonetypevaluetoanothertype.ThisconversioniscalledTypecasting or Type coersion.

֍Thefollowingarevariousinbuilt functionsfortypecasting.

֍ **int():**

1. **int()**
2. **float()**
3. **complex()**
4. **bool()**
5. **str()**

**Note:**

Wecan usethisfunctiontoconvertvaluesfromothertypestoint

1. **>>>int(123.987)**

|  |
| --- |
| **2)123** |
| **3)>>>int(10+5j)** |
| **4)TypeError:can'tconvertcomplextoint** |
| **5)>>>int(True)** |
| **6)1** |
| **7)>>>int(False)** |
| **8)0** |
| **9)>>>int("10")** |
| **10)10** |
| **11)>>>int("10.5")** |
| **12)ValueError:invalidliteralforint()withbase10:'10.5'** |
| **13)>>>int("ten")** |
| **14)ValueError:invalidliteralforint()withbase10:'ten'** |
| **15)>>>int("0B1111")** |
| **16)ValueError:invalidliteralforint()withbase10:'0B1111'** |

* 1. **Wecanconvertfromanytypetointexceptcomplextype.**
  2. **Ifwewanttoconvertstrtypetointtype,compulsarystrshouldcontainonlyintegral value and should be specified in base-10.**



֍ float():

Wecanusefloat()functiontoconvertothertypevaluestofloattype.

* + 1. **>>>float(10)**

|  |
| --- |
| **2)10.0** |
| **3)>>>float(10+5j)** |
| **4)TypeError:can'tconvertcomplexto float** |
| **5)>>>float(True)** |
| **6)1.0** |
| **7)>>>float(False)** |
| **8)0.0** |
| **9)>>>float("10")** |
| **10)10.0** |
| **11)>>>float("10.5")** |
| **12)10.5** |
| **13)>>>float("ten")** |
| **14)ValueError:couldnotconvert stringtofloat: 'ten'** |
| **15)>>>float("0B1111")** |
| **16)ValueError:couldnotconvert stringtofloat: '0B1111'** |

**Note:**

1. **Wecanconvertanytypevaluetofloattypeexceptcomplex type.**
2. **Wheneverwearetryingtoconvertstrtypetofloattypecompulsarystrshouldbe either integral or floating point literal and should be specified only in base-10.**

֍ complex():

Wecanusecomplex()functiontoconvertothertypestocomplextype.

**Form-1:complex(x)**

Wecanusethisfunctiontoconvertxintocomplexnumberwithrealpartxandimaginary part 0.

Eg:

* 1. **complex(10)==>10+0j**

|  |
| --- |
| **2) complex(10.5)===>10.5+0j** |
| **3) complex(True)==>1+0j** |
| **4) complex(False)==>0j** |
| **5) complex("10")==>10+0j** |
| **6) complex("10.5")==>10.5+0j** |
| **7)complex("ten")** |
| **8) ValueError:complex()argisamalformed string** |



**Form-2:complex(x,y)**

Wecanusethismethodtoconvertxandyintocomplexnumbersuchthatxwillbereal part and y will be imaginary part.

Eg: complex(10, -2) 10-2j complex(True,False)1+0j

֍ **bool():**

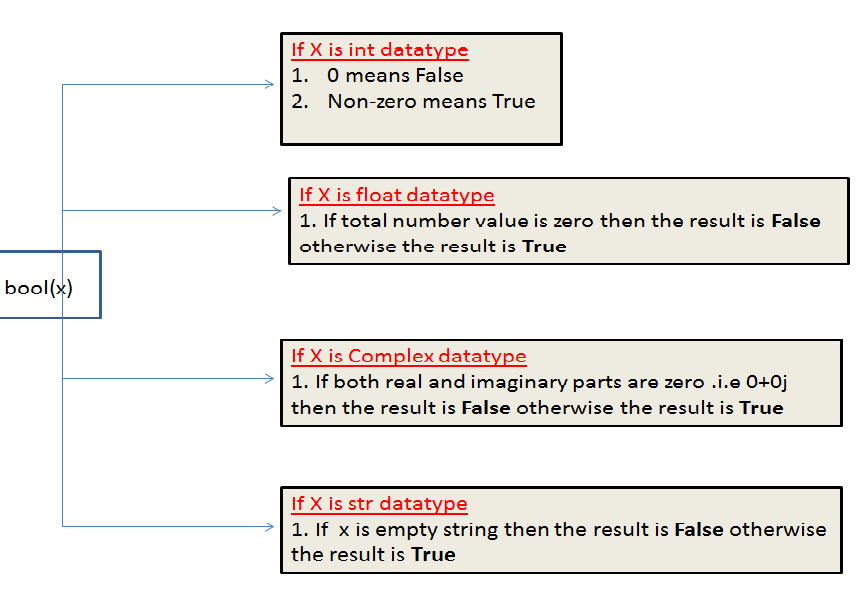
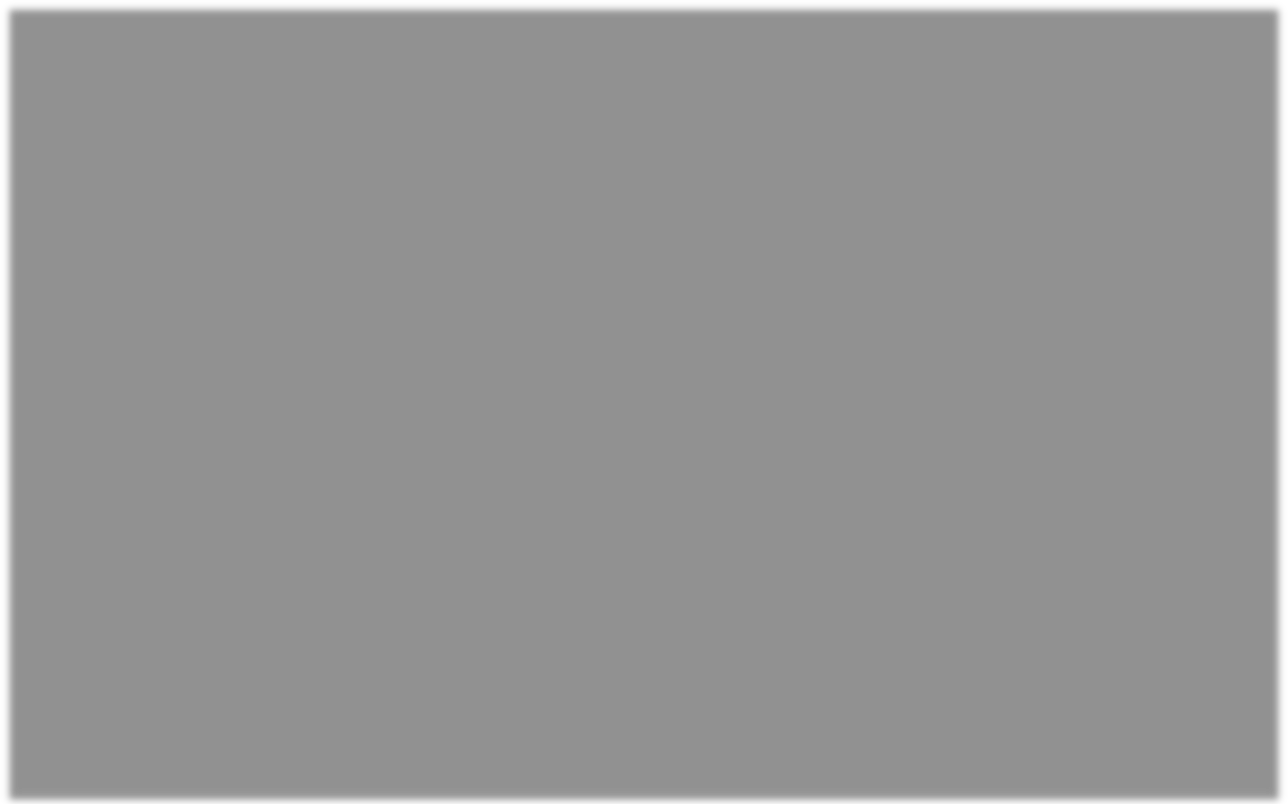
Wecanusethisfunctiontoconvertothertypevaluestobool type.

* + 1. **bool(0)****False**

**2)bool(1)**

**True**

1. **bool(10)****True**
2. **bool(10.5)****True**
3. **bool(0.178)****True**
4. **bool(0.0)****False**
5. **bool(10-2j)****True**
6. **bool(0+1.5j)****True**
7. **bool(0+0j)****False**
8. **bool("True")****True**
9. **bool("False")****True**
10. **bool("")****False**





֍ **str():**

Wecanusethismethodtoconvertothertypevaluestostr type.

1)>>>str(10)

|  |  |
| --- | --- |
| **)** | **'10'** |
| **3)** | **>>>str(10.5)** |
| **4)** | **'10.5'** |
| **5)** | **>>>str(10+5j)** |
| **6)** | **'(10+5j)'** |
| **7)>>>str(True)** | |
| 8)**'True'** | |

FundamentalDataTypesvsImmutability:

֍All Fundamental Data types are immutable. i.e once we creates an object,we cannot performanychangesinthatobject.Ifwearetryingtochange thenwiththosechanges a new object will be created. This non-chageable behaviour is called immutability.

֍In Python if a new object is required, then PVM won’t create object immediately. First it will check is any object available with the required content or not. If available then existing object will be reused. If it is not available then only a new object will be created.Theadvantageofthisapproachismemoryutilizationandperformancewillbe improved.

֍But theprobleminthisapproachis, several referencespointingto thesameobject, by usingonereferenceifweareallowedtochangethecontentintheexistingobjectthen the remaining references will be effected. To prevent this immutability concept is required. According to this once creates an object we are not allowed to change content. If we are trying to change with those changes a new object will be created.

1)>>>a=10

**8)1572353952**

**6)1572353952**

**4)True**

**2)>>>b=10**

3)>>>a is b

5)>>>id(a)

7)>>>id(b)

9)>>>



>>>a=10

>>>b=10

>>> id(a) 1572353952

>>> id(b) 1572353952

>>>aisb

True

>>> a=10+5j

>>>b=10+5j

>>>aisb False

>>> id(a) 15980256

>>>id(b)

15979944

>>>a=True

>>> b=True

>>>aisb True

>>> id(a) 1572172624

>>>id(b)

1572172624

>>>a='durga'

>>>b='durga'

>>>aisb True

>>> id(a) 16378848

>>>id(b)

16378848

1. bytesDataType:

bytesdatatyperepresensagroupofbytenumbersjustlikeanarray.

**1)x =[10,20,30,40]**

**3) type(b)****bytes**

**10**

1. **print(b[-1])****40**
2. **>>>for i in b : print(i) 7)**

**8) 10**

**9) 20**

**10) 30**

**11) 40**

**print(b[0])**

**4)**

**b=bytes(x)**

**2)**

**Conclusion1:**

Theonlyallowedvaluesforbytedatatypeare0to256. Bymistakeifwearetryingto provide any other values then we will get value error.

**Conclusion 2:**

Oncewecreatesbytesdatatypevalue, wecannotchangeitsvalues,otherwisewewillget TypeError.



Eg:

1)>>>x=[10,20,30,40]

**4) TypeError:'bytes'objectdoesnotsupportitemassignment**

**2)>>>b=bytes(x)**

3)>>>b[0]=100

1. bytearrayDataType:

bytearrayisexactlysameasbytesdatatypeexceptthatitselementscanbe modified.

Eg1:

1)x=[10,20,30,40]

**12)30**

**10)100**

**8) b[0]=100**

**6)30**

**4)10**

**2)b=bytearray(x)**

3)for iinb:print(i)

5)20

7)40

9) foriinb:print(i)

11)20

13)40

Eg 2:

1)>>>x =[10,256]

**2)>>>b=bytearray(x)**

3)ValueError: byte mustbeinrange(0,256)

1. ListDataType:

Ifwewanttorepresentagroupofvaluesasasingleentitywhereinsertionorder required to preserve and duplicates are allowed then we should go for list data type.

1. **InsertionOrderispreserved**
2. **HeterogeneousObjectsareallowed**
3. **Duplicatesare allowed**
4. **Growableinnature**
5. **Valuesshouldbeenclosedwithinsquarebrackets.**



**Eg:**

**1)list=[10,10.5,'durga',True,10]**

**2) print(list)#[10,10.5,'durga',True,10]**

**Eg:**

1)list=[10,20,30,40]

|  |
| --- |
| **2)>>>list[0]** |
| **3)10** |
| **4)>>> list[-1]** |
| **5)40** |
| **6)>>>list[1:3]** |
| **7)[20, 30]** |
| **8)>>>list[0]=100** |
| **9)>>>for iinlist:print(i)** |
| **10)...** |
| **11)100** |
| **12)20** |
| **13)30** |
| **14)40** |

listisgrowableinnature.i.ebasedonourrequirementwecanincreaseordecreasethe size.

1)>>>list=[10,20,30]

|  |
| --- |
| **2)>>>list.append("durga")** |
| **3)>>>list** |
| **4)[10, 20, 30, 'durga']** |
| **5)>>>list.remove(20)** |
| **6)>>>list** |
| **7)[10, 30, 'durga']** |
| **8)>>>list2=list\*2** |
| **9)>>>list2** |
| **10)[10,30,'durga', 10,30,'durga']** |

Note:Anordered,mutable,heterogenouscollectionofeleemntsisnothingbutlist, where duplicates also allowed.



1. TupleDataType:
   * **tupledatatypeisexactlysameaslistdatatypeexceptthatitisimmutable.i.ewe cannot chage values.**
   * **Tupleelementscanberepresentedwithinparenthesis.**

**Eg:**

1) t=(10,20,30,40)

**8)>>>t.remove(10)**

**6)>>>t.append("durga")**

**4) t[0]=100**

**2) type(t)**

3)<class'tuple'>

5)TypeError:'tuple'objectdoesnotsupport item assignment

7)AttributeError:'tuple'objecthasnoattribute'append'

9)AttributeError:'tuple'objecthasnoattribute'remove'

Note:tupleisthereadonlyversionoflist

1. RangeDataType:
   * **rangeDataTyperepresentsasequenceof numbers.**
   * **TheelementspresentinrangeDatatypearenotmodifiable.i.erangeDatatypeis immutable.**

**Form-1:range(10)**

generatenumbersfrom0to9

**Eg:**

r=range(10)

foriinr: print(i) 0to9

**Form-2:range(10,20)**

generatenumbersfrom10to19

**Eg:**

r=range(10,20)

foriinr: print(i) 10to19



Form-3:range(10,20,2) 2 means increment value

**Eg:**

r=range(10,20,2)

foriinr:print(i)10,12,14,16,18

Wecan access elementspresentintherangeDataTypebyusingindex.

**Eg:**

r=range(10,20) r[0] 10

r[15]IndexError:rangeobjectindexoutofrange We cannot modify the values of range data type

**Eg:**

r[0]= 100

TypeError:'range'objectdoesnotsupportitemassignment We can create a list of values with range data type

**Eg:**

1)>>>l=list(range(10))

**2)>>>l**

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

1. setDataType:

֍Ifwewanttorepresentagroupofvalueswithoutduplicateswhereorderisnot important then we should go for set Data Type.

1. **Insertionorderisnotpreserved**
2. **Duplicatesarenotallowed**
3. **Heterogeneousobjectsareallowed**
4. **Indexconceptisnot applicable**
5. **Itismutable collection**
6. **Growableinnature**

**26**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**Eg:**

1) s={100,0,10,200,10,'durga'}

**2) s# {0, 100,'durga', 200,10}**

3)s[0]TypeError:'set'objectdoesnotsupport indexing

֍setisgrowableinnature,basedonourrequirementwecanincreaseordecreasethe size.

1)>>>s.add(60)

|  |
| --- |
| **2)>>>s** |
| **3){0,100, 'durga', 200,10,60}** |
| **4)>>>s.remove(100)** |
| **5)>>>s** |
| **6){0, 'durga',200,10,60}** |

1. frozensetDataType:

֍It is exactlysameas setexcept thatitis immutable.

֍Hencewecannotuseaddorremove functions.

1)>>>s={10,20,30,40}

**16)>>>fs.remove(10)**

**14)>>>fs.add(70)**

**12)30**

**10)10**

**8)...**

**6)frozenset({40, 10,20,30})**

**4)<class'frozenset'>**

**2)>>>fs=frozenset(s)**

3)>>>type(fs)

5)>>>fs

7)>>>for iinfs:print(i)

9)40

11)20

13)

15)AttributeError:'frozenset'objecthasnoattribute'add'

17)AttributeError:'frozenset'objecthasnoattribute'remove'



1. dictDataType:

|  |  |  |
| --- | --- | --- |
|  | ֍ | **Ifwewanttorepresentagroupof valuesaskey-valuepairsthenweshould gofor**  **dictdata type.** |
| ֍ | **Eg:d = {101:'durga',102:'ravi',103:'shiva'}** |
| ֍ | **Duplicate keys are not allowed but values can be duplicated. If we are trying to insertanentrywithduplicatekeythenoldvaluewillbe replacedwithnewvalue.** |
| **Eg:** |  |  |
|  | **1)** | **>>>d={101:'durga',102:'ravi',103:'shiva'}** |
|  | **2)** | **>>>d[101]='sunny'** |
|  | **3)** | **>>>d** |
|  | **4)** | **{101:'sunny',102:'ravi',103:'shiva'}** |
|  | **5)** |  |
|  | **6)** | **Wecancreateemptydictionaryasfollows** |
|  | **7)** | **d={ }** |
|  | **8)** | **Wecanaddkey-valuepairsas follows** |
|  | **9)** | **d['a']='apple'** |
|  | 1. **d['b']='banana'** 2. **print(d)** | |

Note:dictismutableandtheorderwon’tbe preserved.

**Note:**

1. **Ingeneralwecanusebytesandbytearraydatatypestorepresentbinaryinformation like images, video files etc**
2. **InPython2longdatatypeisavailable.Butin Python3itisnotavailableandwecan represent long values also by using int type only.**
3. **InPythonthereisnochardatatype.Hencewecanrepresentcharvaluesalsobyusing str type.**



##### SummaryofDatatypesinPython3

|  |  |  |  |
| --- | --- | --- | --- |
| **Datatype** | **Description** | **IsImmutable?** | **Example** |
| **Int** | **We can use to represent the whole/integral numbers** | **Immutable** | **>>>a=10**  **>>>type(a)**  **<class'int'>** |
| **Float** | **We can use to represent the decimal/floating**  **point numbers** | **Immutable** | **>>> b=10.5**  **>>>type(b)**  **<class'float'>** |
| **Complex** | **We can use to represent the complexnumbers** | **Immutable** | **>>> c=10+5j**  **>>>type(c)**  **<class'complex'>**  **>>>c.real 10.0**  **>>>c.imag 5.0** |
| **Bool** | **We can use to representthelogical values(Onlyallowed values are True and**  **False)** | **Immutable** | **>>>flag=True**  **>>>flag=False**  **>>>type(flag)**  **<class'bool'>** |
| **Str** | **Torepresent sequence of Characters** | **Immutable** | **>>>s='durga'**  **>>>type(s)**  **<class'str'>**  **>>>s="durga"**  **>>> s='''Durga Software Solutions...Ameerpet'''**  **>>>type(s)**  **<class'str'>** |
| **bytes** | **To represent a sequence of byte valuesfrom0-255** | **Immutable** | **>>>list=[1,2,3,4]**  **>>>b=bytes(list)**  **>>>type(b)**  **<class'bytes'>** |
| **bytearray** | **To represent a sequence of byte valuesfrom0-255** | **Mutable** | **>>>list=[10,20,30]**  **>>>ba=bytearray(list)**  **>>>type(ba)**  **<class'bytearray'>** |
| **range** | **Torepresentarange of values** | **Immutable** | **>>>r=range(10)**  **>>> r1=range(0,10)**  **>>>r2=range(0,10,2)** |
|  |  |  |  |



|  |  |  |  |
| --- | --- | --- | --- |
| **list** | **To represent an orderedcollectionof objects** | **Mutable** | **>>>l=[10,11,12,13,14,15]**  **>>>type(l)**  **<class'list'>** |
| **tuple** | **To represent an orderedcollectionsof objects** | **Immutable** | **>>>t=(1,2,3,4,5)**  **>>>type(t)**  **<class'tuple'>** |
| **set** | **To represent an unorderedcollection of unique objects** | **Mutable** | **>>>s={1,2,3,4,5,6}**  **>>>type(s)**  **<class'set'>** |
| **frozenset** | **To represent an unorderedcollection of unique objects** | **Immutable** | **>>>s={11,2,3,'Durga',100,'Ramu'}**  **>>> fs=frozenset(s)**  **>>>type(fs)**  **<class'frozenset'>** |
| **dict** | **Torepresentagroup of key value pairs** | **Mutable** | **>>>**  **d={101:'durga',102:'ramu', 103:'hari'}**  **>>>type(d)**  **<class'dict'>** |

1. NoneDataType:
   * **NonemeansnothingorNo value associated.**
   * **Ifthevalueisnotavailable,thentohandlesuchtypeofcasesNoneintroduced.**
   * **ItissomethinglikenullvalueinJava.**

**Eg:**

defm1(): a=10

print(m1()) None



EscapeCharacters:

InStringliteralswecanuseesacpecharacterstoassociateaspecial meaning.

1)>>>s="durga\nsoftware"

**14)>>>print(s)**

**12)SyntaxError:invalid syntax**

**10) s="Thisis"symbol"**

**8)>>>s="Thisis "symbol"**

**6)>>>print(s)**

**4)software**

**2)>>>print(s)**

3)durga

5)>>>s="durga\tsoftware"

7)durgasoftware

9) File"<stdin>",line1

11) ^

13)>>>s="Thisis \"symbol"

15)Thisis"symbol

ThefollowingarevariousimportantescapecharactersinPython

1. **\n**
2. **\t**
3. **\r**
4. **\b**
5. **\f**
6. **\v**
7. **\'**
8. **\"**
9. **\\**

**....**

**NewLine**

HorizontalTab

Carriage Return

BackSpace

FormFeed

VerticalTab

SingleQuote

DoubleQuote

BackSlash Symbol

Constants:

* **ConstantsconceptisnotapplicableinPython.**
* **Butitisconventionto useonlyuppercasecharactersifwedon’twanttochange value.**
* **MAX\_VALUE = 10**
* **Itisjustconventionbutwecanchangethe value.**



### OPERATORS



* **Operatorisasymbolthatperformscertain operations.**
* **Python providesthefollowingsetof operators**
  1. **ArithmeticOperators**
  2. **RelationalOperatorsORComparisonOperators**
  3. **Logicaloperators**
  4. **Bitwiseoeprators**
  5. **Assignmentoperators**
  6. **Specialoperators**

1. ArithmeticOperators:

|  |  |  |
| --- | --- | --- |
| **1)+** |  | **Addition** |
| **2)–** |  | **Subtraction** |
| **3)\*** |  | **Multiplication** |
| **4)/** |  | **DivisionOperator** |
| **5)%** |  | **ModuloOperator** |
| **6)//** |  | **FloorDivisionOperator** |
| **7)\*\*** |  | **ExponentOperatorORPowerOperator** |

**Eg:test.py**

* 1. **a=10**

**8)print('a%b=',a%b)**

**6)print('a/b=',a/b)**

**4)print('a-b=',a-b)**

**2)b=2**

3)print('a+b=',a+b)

5)print('a\*b=',a\*b)

7)print('a//b=',a//b)

9)print('a\*\*b=',a\*\*b)

**Output:**

Pythontest.pyORpytest.py a+b = 12

a-b= 8

a\*b=20

a/b=5.0



a//b=5

a%b= 0

a\*\*b=100

**Eg:**

1)a =10.5

|  |  |
| --- | --- |
| **2)** | **b=2** |
| **3)** | |
| **4)** | **a+b=12.5** |
| **5)** | **a-b= 8.5** |
| **6)** | **a\*b=21.0** |
| **7)** | **a/b=5.25** |
| **8)** | **a//b=5.0** |
| **9)** | **a%b= 0.5** |
| **10)a\*\*b=110.25** | |

**Eg:**

10/25.0

**10//2****5**

10.0/25.0

10.0//25.0

**Note:**

֍/operatoralwaysperformsfloatingpointarithmetic.Henceitwillalwaysreturnsfloat value.

֍But Floor division (//) can perform both floating point and integral arithmetic. If argumentsareinttypethenresultisinttype.Ifatleastoneargumentisfloattypethen result is float type.

**Note:**

֍Wecan use+,\*operatorsfor strtype also.

֍Ifwewanttouse+operatorforstrtypethencompulsorybothargumentsshouldbe str type only otherwise we will get error.

1)>>>"durga"+10

**4)'durga10'**

**2)TypeError:mustbestr,notint**

3)>>>"durga"+"10"

֍Ifweuse\*operatorforstrtypethencompulsoryoneargument shouldbeintand other argument should be str type.



֍2\*"durga""durga"\*2

2.5\*"durga"TypeError: can't multiply sequence by non-int of type 'float'"durga"\*"durga"TypeError:can'tmultiplysequencebynon-intoftype'str'

֍+StringConcatenation Operator

֍\*StringMultiplication Operator

**Note:Foranynumberx,**

x/0andx%0alwaysraises"ZeroDivisionError"

10/0

10.0/0

**.....**

1. **RelationalOperators:>,>=,<,<=**
   1. **a=10**

**10)a <bisTrue**

**8)a >bisFalse**

**6)print("a<=bis",a<=b)**

**4)print("a>=bis",a>=b)**

**2)b=20**

3)print("a>bis",a>b)

5)print("a<bis",a<b)

7)

9)a >=bisFalse

11)**a<=bisTrue**

Wecanapplyrelationaloperatorsforstrtypesalso.

**Eg2:**

1)a="durga"

**10)a <bisFalse**

**8)a >bisFalse**

**6)print("a<=bis ",a<=b)**

**4)print("a>=bis ",a>=b)**

**2)b="durga"**

3)print("a>bis ",a>b)

5)print("a<bis ",a<b)

7)

9)a >=bisTrue

11)a <=bisTrue



**Eg:**

1)print(True>True)False

**6)print(10>'durga')**

**4)print(False>True)False**

**2)print(True>=True)True**

3)print(10>True)True

5)

7) TypeError:'>'notsupportedbetweeninstancesof'int'and'str'

**Eg:**

1)a=10

|  |
| --- |
| **2)b=20** |
| **3)if(a>b):** |
| **4) print("aisgreaterthanb")** |
| **5)else:** |
| **6) print("aisnotgreaterthan b")** |

**Output:aisnotgreaterthan b**

Note:Chaining of relational operators is possible. In the chaining, if all comparisons returnsTruethenonlyresultis True.IfatleastonecomparisonreturnsFalsethenthe result is False

1)10<20True

**4)10<20<30<40>50****False**

**2)10<20<30****True**

3)10<20<30<40True

1. EqualityOperators:==,!=

Wecanapplytheseoperatorsfor anytypeevenforincompatibletypes also.

1)>>>10==20

**10)True**

**8)True**

**6)False**

**4)True**

**2)False**

3)>>>10!=20

5)>>>10==True

7)>>>False==False

9)>>>"durga"=="durga"

11)>>>10=="durga"



12)**False**

Note:Chainingconceptisapplicableforequalityoperators.Ifatleastonecomparison returns False then the result is False. Otherwise the result is True.

1)>>>10==20==30==40

**4)True**

**2)False**

3)>>>10==10==10==10

1. LogicalOperators:and,or,not

Wecan applyforall types.

* + **ForbooleanTypes Behaviour:**

andIfbothargumentsareTruethenonlyresultisTrue orIf atleast one arugemnt is True then result is True not Complement

TrueandFalseFalse True or False True not False True

* + **Fornon-booleanTypes Behaviour:**

0meansFalse

non-zeromeans True

emptystringisalwaystreatedas False

**xandy:**

Ifxisevaluatestofalsereturnxotherwisereturny Eg:

10 and20

0 and20

Iffirstargumentiszero thenresultiszerootherwiseresultisy

**xor y:**

IfxevaluatestoTruethen resultisxotherwiseresultisy

10 or20 10

0 or2020



**not x:**

IfxisevalutatestoFalsethenresultisTrueotherwiseFalse

not10False not 0 True

**Eg:**

1)"durga"and"durgasoft"==>durgasoft

**6)not""==>True**

**4)"" or"durga"==>"durga"**

**2)""and"durga"==>""**

3)"durga"and ""==>""

5)"durga"or""==>"durga"

7)not"durga"==>False

1. BitwiseOperators:

֍Wecan applytheseoperators bitwise.

֍Theseoperatorsare applicableonlyforintandbooleantypes.

֍Bymistakeifwearetryingtoapplyfor anyothertypethenwewillget Error.

֍**&, |,^, ~,<<,>>**

֍print(4&5)Valid

֍print(10.5 &5.6)

TypeError:unsupportedoperandtype(s)for&:'float'and'float'

֍print(True&True) Valid

֍&Ifboth bitsare1then onlyresultis1otherwiseresultis0

֍|If atleastonebitis1thenresultis1otherwiseresultis0

֍^Ifbits aredifferentthen onlyresultis1otherwiseresultis0

֍~bitwisecomplement operator

֍**1****0&0****1**

֍<<BitwiseLeftShift

֍>>BitwiseRightShift

֍**print(4&5)****4**

֍**print(4|5)****5**

֍**print(4^5)****1**



|  |  |
| --- | --- |
| **Operator** | **Description** |
| **&** | **Ifbothbitsare1 thenonlyresultis1 otherwiseresultis0** |
| **|** | **Ifatleastonebitis1thenresultis1 otherwiseresultis0** |
| **^** | **Ifbitsaredifferentthenonlyresultis1otherwiseresultis0** |
| **~** | **bitwisecomplementoperatori.e1means0and0means1** |
| **>>** | **BitwiseLeftshiftOperator** |
| **<<** | **BitwiseRightshift Operator** |

BitwiseComplementOperator (~):

Wehavetoapplycomplementfortotalbits. Eg: print(~5)-6

**Note:**

֍Themostsignificantbitactsassignbit.0valuerepresents +venumberwhereas1 represents -ve value.

֍Positivenumberswillberepesenteddirectlyinthememorywhereas -venumberswill be represented indirectly in 2's complement form.

1. ShiftOperators:

**<< Left ShiftOperator**

Aftershiftingtheemptycellswehavetofillwithzero

print(10<<2)40

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **1** | **0** | **1** | **0** | **0** | **0** |

**>>RightShift Operator**

Aftershiftingtheemptycellswehavetofillwithsignbit.(0for +veand1for-ve)

print(10>>2)2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **0** | **0** | **1** | **0** | **1** | **0** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | **0** | **0** | **0** | **0** | **0** | **1** | **0** |



Wecanapplybitwiseoperatorsforbooleantypesalso

֍print(True&False)False

֍print(True| False)True

֍print(True^False)True

֍**print(~True)****-2**

֍print(True<<2)4

֍print(True>>2)0

1. AssignmentOperators:

֍Wecan useassignmentoperatortoassignvaluetothevariable.

Eg:x=10

֍Wecancombineasignmentoperatorwithsomeotheroperatortoformcompound assignment operator.

Eg:x+=10x =x+10

Thefollowingisthelistofallpossiblecompound assignmentoperatorsinPython.

 **+=**

* **-=**
* **\*=**
* **/=**
* **%=**
* **//=**
* **\*\*=**
* **&=**
* **|=**
* **^=**
* **>>=**
* **<<=**

**Eg:**

1)x=10

**2)x+=20**

3)print(x)30

**Eg:**

1)x=10

**2)x&=5**

3)print(x)0



1. TernaryOperatorORConditionalOperator

Syntax:x=firstValueifconditionelse secondValue

IfconditionisTruethenfirstValuewillbeconsideredelsesecondValuewillbeconsidered.

**Eg1:**

1)a,b=10,20

**2)x=30ifa<b else40**

3)print(x) #30

Eg2:Readtwonumbersfromthekeyboardandprintminimum value

1)a=int(input("Enter FirstNumber:"))

**4)print("Minimum Value:",min)**

**2)b=int(input("EnterSecond Number:"))**

3)min=aifa<belseb

**Output:**

Enter First Number:10 EnterSecondNumber:30 Minimum Value: 10

Note:NestingofTernaryOperatoris Possible.

**Q)ProgramforMinimumof3Numbers**

1)a=int(input("Enter FirstNumber:"))

**4)min=aifa<b and a<celsebif b<c elsec**

**2)b=int(input("EnterSecond Number:"))**

3)c=int(input("EnterThirdNumber:"))

5)print("Minimum Value:",min)

**Q)ProgramforMaximumof3Numbers**

1)a=int(input("Enter FirstNumber:"))

**4)max=aif a>banda>celsebifb>celsec**

**2)b=int(input("EnterSecond Number:"))**

3)c=int(input("EnterThirdNumber:"))

5)print("MaximumValue:",max)

**41**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**Eg:**

1)a=int(input("Enter FirstNumber:"))

**2)b=int(input("EnterSecond Number:"))**

3)print("Bothnumbersareequal"ifa==belse"FirstNumberisLessthanSecondNu mber"if a<b else "First Number Greater than Second Number")

Output:

**D:\python\_classes>pytest.py Enter First Number:10**

**EnterSecondNumber:10 Both numbers are equal**

**D:\python\_classes>pytest.py Enter First Number:10**

**EnterSecondNumber:20**

**FirstNumberisLessthanSecondNumber**

**D:\python\_classes>pytest.py Enter First Number:20**

**EnterSecondNumber:10**

**FirstNumberGreaterthanSecondNumber**

1. SpecialOperators:

Python definesthefollowing2special operators

* 1. **Identity Operators**
  2. **Membershipoperators**

1. IdentityOperators
   * **Wecanuseidentityoperatorsfor addresscomparison.**
   * **Thereare2identityoperatorsare available**
2. **is**
3. **is not**
   * **r1isr2returnsTrueifbothr1andr2arepointingtothesameobject.**
   * **r1isnotr2returnsTrueifbothr1 andr2arenotpointingtothesameobject.**

**Eg:**

1) a=10

**4)x=True**

**2)b=10**

3)print(a isb) True

**42**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**5)y=True**

**6)print( xis y) True**

|  |  |  |
| --- | --- | --- |
| **Eg:** |  | |
|  | **1)** | **a="durga"** |
|  | **2)** | **b="durga"** |
|  | **3)** | **print(id(a))** |
|  | **4)** | **print(id(b))** |
|  | **5)** | **print(aisb)** |
| **Eg:** |  |  |
|  | **1)** | **list1=["one","two","three"]** |
|  | **2)** | **list2=["one","two","three"]** |
|  | **3)** | **print(id(list1))** |
|  | **4)** | **print(id(list2))** |
|  | **5)** | **print(list1islist2)False** |
|  | **6)** | **print(list1isnotlist2)True** |
|  | 7) | **print(list1== list2) True** |

Note:Wecanuseisoperatorforaddresscomparisonwhereas==operatorforcontent comparison.

1. MembershipOperators:
   * **WecanuseMembershipoperatorstocheckwhetherthegivenobjectpresent inthe given collection. (It may be String, List, Set, Tuple OR Dict)**
   * **In****ReturnsTrueifthegivenobjectpresentin thespecified Collection**
   * **notin****RetrunsTrueifthegiven objectnotpresentinthespecified Collection**

**Eg:**

|  |  |  |
| --- | --- | --- |
|  | **1)** | **x="hellolearningPythonisvery easy!!!"** |
| **2)** | **print('h'inx) True** |
| **3)** | **print('d'inx) False** |
| **4)** | **print('d'notinx) True** |
| **5)** | **print('Python'in x) True** |
| **Eg:** | **1)** | **list1=["sunny","bunny","chinny","pinny"]** |
|  | **2)** | **print("sunny"inlist1)True** |
|  | **3)** | **print("tunny"inlist1)False** |
|  | **4)** | **print("tunny"notinlist1)True** |



OperatorPrecedence:

Ifmultipleoperatorspresentthenwhichoperatorwillbeevaluatedfirstisdecidedby operator precedence.

**Eg:**

print(3+10\*2)23

print((3+10)\*2)26

ThefollowinglistdescribesoperatorprecedenceinPython

1. **()****Parenthesis**
2. **\*\*****Exponential Operator**
3. **~,-****BitwiseComplementOperator,UnaryMinusOperator**
4. **\*,/,%,//****Multiplication,Division,Modulo,Floor Division**

5)+,-Addition, Subtraction

1. **<<, >>****Left andRightShift**
2. **&****BitwiseAnd**
3. **^****BitwiseX-OR**
4. **|****BitwiseOR**
5. **>,>=,<,<=,==,!=****RelationalORComparisonOperators**

11)=,+=,-=, \*=...Assignment Operators

1. **is,isnot****Identity Operators**
2. **in,not in****Membership operators**
3. **not****Logicalnot**
4. **and****Logicaland**
5. **or****Logicalor**

1)a=30

**13)6.0+3+8.0-2**

**11)3/2\*4+3+8.0-2**

**9)3/2\*4+3+(10/5)\*\*3-2**

1. **print((a+b)\*(c/d))****100.0**
2. **print(a+(b\*c)/d)****70.0**

**4)d=5**

**2)b=20**

3)c=10

5)print((a+b)\*c/d)100.0

8)

10)3/2\*4+3+2.0\*\*3-2

12)1.5\*4+3+8.0-2

14)15.0

**44**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**45**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

**MathematicalFunctions(mathModule)**

֍AModuleiscollectionoffunctions, variablesandclasses etc.

֍math isamodulethatcontainsseveralfunctionstoperformmathematicaloperations.

֍Ifwewanttouseanymodulein Python,firstwehavetoimportthat module. import math

֍Onceweimport amodulethenwecancallanyfunctionofthatmodule.

1)importmath

**2)print(math.sqrt(16))**

3)print(math.pi)

**Output**

4.0

3.141592653589793

֍Wecancreatealiasnamebyusingaskeyword.importmathasm

֍Oncewecreatealiasname,byusingthatwecanaccessfunctions andvariablesofthat module.

1)import mathas m

**2)print(m.sqrt(16))**

3)print(m.pi)

֍Wecanimportaparticularmember ofamoduleexplicitlyasfollows

from math import sqrt frommathimportsqrt,pi

֍Ifweimport amemberexplicitlythenitisnotrequiredtousemodulenamewhile accessing.

1)from mathimportsqrt,pi

**4)print NameError:name****(math.pi)'math'isnotdefined**

**2)print(sqrt(16))**

3)print(pi)



ImportantFunctionsofmathModule:

1. **ceil(x)**
2. **floor(x)**
3. **pow(x,y)**
4. **factorial(x)**
5. **trunc(x)**
6. **gcd(x,y)**
7. **sin(x)**
8. **cos(x)**
9. **tan(x)**
10. **....**

ImportantVariablesofmathModule:

pi3.14

**e****2.71**

infinfinity

nannot anumber

1. **WriteaPython ProgramtofindAreaofCircle pi\*r\*\*2**

1)from mathimportpi

**2)r=16**

3)print("Areaof Circleis:",pi\*r\*\*2)

Output:AreaofCircleis:804.247719318987



INPUTANDOUTPUTSTATEMENTS

ReadingDynamicInputfromtheKeyboard:

InPython2thefollowing2functionsareavailabletoreaddynamicinputfromthe keyboard.

1. **raw\_input()**
2. **input()**
   1. **raw\_input():**

Thisfunctionalwaysreadsthedatafromthekeyboardintheformof StringFormat. Wehavetoconvertthatstringtypetoourrequiredtypebyusingthecorresponding type casting methods.

Eg:x=raw\_input("EnterFirstNumber:")

print(type(x))Itwillalwaysprintstrtypeonlyfor anyinputtype

* 1. input():

input()functioncanbeusedtoreaddatadirectlyinourrequiredformat.Wearenot required to perform type casting.

x=input("EnterValue)type(x)

10 int "durga"str

10.5float Truebool

**\*\*\*Note:**

* **Butin Python 3wehaveonlyinput()methodand raw\_input()methodisnot available.**
* **Python3input()functionbehaviourexactlysameasraw\_input()methodofPython2.**

i.eeveryinputvalueistreatedasstrtypeonly.

* **raw\_input()functionof Python 2isrenamedasinput()functioninPython3.**

**1) >>>type(input("Entervalue:"))**

**3)<class'str'>**

**5)Entervalue:10.5**

**7)**

**6)<class'str'>**

**4)**

**2)Entervalue:10**



9)<class'str'>

**8)Entervalue:True**

**Q)Writeaprogramtoread2numbersfromthekeyboardandprintsum**

1)x=input("Enter FirstNumber:")

**4)j=int(y)**

**2)y=input("EnterSecondNumber:")**

3)i=int(x)

5)print("The Sum:",i+j)

Enter First Number: 100 EnterSecondNumber:200 The Sum: 300

**-----------------------------------------------------**

1)x=int(input("EnterFirstNumber:"))

**2)y=int(input("Enter Second Number:"))**

3)print("The Sum:",x+y)

**-----------------------------------------------------------**

print("TheSum:",int(input("EnterFirstNumber:"))+int(input("EnterSecondNumber:")))

**Q)WriteaProgramtoreadEmployeeDatafromtheKeyboardandprint that Data**

1)eno=int(input("EnterEmployee No:"))

**10)print("EmployeeAddress :",eaddr)**

**8)print("EmployeeName:",ename)**

**6)print("PleaseConfirmInformation")**

**4)eaddr=input("EnterEmployeeAddress:")**

**2)ename=input("EnterEmployee Name:")**

3)esal=float(input("EnterEmployeeSalary:"))

5)married=bool(input("EmployeeMarried ?[True|False]:"))

7)print("EmployeeNo:",eno)

9)print("EmployeeSalary :",esal)

11)print("EmployeeMarried?:",married)

D:\Python\_classes>pytest.py Enter Employee No:100

Enter Employee Name:Sunny Enter Employee Salary:1000 EnterEmployeeAddress:Mumbai



EmployeeMarried?[True|False]:True Please Confirm Information

Employee No : 100 Employee Name : Sunny Employee Salary : 1000.0 EmployeeAddress:Mumbai Employee Married ? : True

Howtoreadmultiplevaluesfromthekeyboardinasingle line:

**1)a,b= [int(x)forxininput("Enter2 numbers:").split()]**

**2)print("Product is:",a\*b)**

D:\Python\_classes>pytest.py Enter 2 numbers :10 20 Product is : 200

Note: split() function can take space as seperator by default .Butwecanpass anything as seperator.

**Q)Writeaprogramtoread3floatnumbersfromthekeyboardwith, seperator and print their sum**

**1)a,b,c=[float(x)for xininput("Enter3 floatnumbers:").split(',')]**

**2)print("TheSum is:",a+b+c)**

D:\Python\_classes>pytest.py

Enter3floatnumbers:10.5,20.6,20.1 The Sum is : 51.2

eval():

evalFunctiontakeaStringandevaluatethe Result.

Eg:x = eval(“10+20+30”)

print(x) Output:60

Eg:x=eval(input(“EnterExpression”)) Enter Expression:10+2\*3/4

Output:11.5



eval()canevaluatetheInputtolist,tuple,set,etcbasedtheprovidedInput.

Eg:WriteaProgramtoaccept listfromthekeynboardonthedisplay

1)l=eval(input(“EnterList”))

**2)print(type(l))**

3)print(l)

COMMAND LINE ARGUMENTS

* **argvisnotArrayitisaList.ItisavailablesysModule.**
* **TheArgumentwhicharepassingatthetimeofexecutionarecalledCommandLine Arguments.**

Eg:D:\Python\_classespytest.py102030

CommandLineArguments

WithinthePythonProgramthisCommandLineArgumentsareavailableinargv.Whichis present in SYS Module.

|  |  |  |  |
| --- | --- | --- | --- |
| **test.py** | **10** | **20** | **30** |

Note:argv[0]representsNameofProgram.Butnotfirst CommandLineArgument. argv[1] represent First Command Line Argument.

**Program:Tochecktypeof argvfromsys**

import argvprint(type(argv))

D:\Python\_classes\pytest.py

WriteaProgramtodisplayCommandLineArguments

1)from sysimportargv

|  |
| --- |
| **2)print(“TheNumber ofCommandLineArguments:”,len(argv))** |
| **3)print(“TheList ofCommandLineArguments:”,argv)** |
| **4)print(“CommandLineArguments onebyone:”)** |
| **5)forxin argv:** |
| **6) print(x)** |



D:\Python\_classes>pytest.py102030

TheNumberofCommand LineArguments: 4

TheListofCommandLineArguments:[‘test.py’,‘10’,’20’,’30’] Command Line Arguments one by one:

test.py 10

20

30

**---------------------------**

1)from sysimportargv

**6) sum=sum+n**

**4)forxin args:**

**2)sum=0**

3)args=argv[1:]

5) n=int(x)

7)print("The Sum:",sum)

D:\Python\_classes>pytest.py10203040

The Sum:100

Note 1:Usually space is seperator between command line arguments. If our command lineargumentitselfcontainsspacethenweshouldenclosewithindoublequotes(butnot single quotes)

**1)from sysimportargv**

**2)print(argv[1])**

D:\Python\_classes>pytest.pySunnyLeone Sunny

D:\Python\_classes>pytest.py'SunnyLeone''Sunny

D:\Python\_classes>pytest.py"SunnyLeone"Sunny Leone

Note2:WithinthePythonprogramcommandlineargumentsareavailableintheString form. Based on our requirement, we can convert into corresponding type by using type casting methods.

1)from sysimportargv

**2)print(argv[1]+argv[2])**

3)print(int(argv[1])+int(argv[2]))



D:\Python\_classes>pytest.py1020

1020

30

Note3:Ifwearetryingtoaccesscommandlineargumentswithoutofrangeindexthen we will get Error.

**1)from sysimportargv**

**2)print(argv[100])**

D:\Python\_classes>pytest.py1020 IndexError: list index out of range

Note:InPythonthereisargparsemoduletoparsecommandlineargumentsanddisplay some help messages whenever end user enters wrong input.

input() raw\_input()

CommandLineArguments

OutputStatements:

Wecanuseprint()functiontodisplay output.

Form-1:print()withoutanyargument Just it prints new line character

**Form-2:**

1)print(String):

|  |
| --- |
| **2)print("Hello World")** |
| **3)Wecanuse escapecharactersalso** |
| **4)print("Hello \n World")** |
| **5)print("Hello\tWorld")** |
| **6)Wecan userepetetionoperator (\*) in the string** |
| **7)print(10\*"Hello")** |
| **8)print("Hello"\*10)** |
| **9)Wecanuse+operatoralso** |
| **10)print("Hello"+"World")** |



**Note:**

֍IfbothargumentsareStringtypethen+ operatoractsasconcatenationoperator.

֍If oneargumentisstringtypeandsecondisanyothertypelikeintthenwewillget Error.

֍Ifbothargumentsarenumbertypethen+operatoractsasarithmeticaddition operator.

**Note:**

**1)print("Hello"+"World")**

**2)print("Hello","World")**

HelloWorld HelloWorld

Form-3:print()withvariablenumberofarguments

**1)a,b,c=10,20,30**

**2)print("TheValuesare:",a,b,c)**

**Output:TheValuesare:102030**

Bydefaultoutputvaluesareseperatedbyspace.Ifwewantwecanspecifyseperatorby using "sep" attribute

1)a,b,c=10,20,30

**2)print(a,b,c,sep=',')**

3)print(a,b,c,sep=':')

D:\Python\_classes>pytest.py 10,20,30

10:20:30

**Form-4:print()withendattribute**

1)print("Hello")

**2)print("Durga")**

3)print("Soft")

**Output:Hello Durga Soft**



Ifwewantoutputinthesamelinewithspace

1)print("Hello",end='')

**2)print("Durga",end='')**

**3)print("Soft")**

**Output:HelloDurgaSoft**

Note:Thedefault valueforendattributeis\n,whichisnothingbutnewline character.

**Form-5:print(object) statement**

Wecanpassanyobject(likelist, tuple,setetc)asargumenttotheprint()statement.

1)l=[10,20,30,40]

**4)print(t)**

**2)t=(10,20,30,40)**

3)print(l)

**Form-6:print(String,variablelist)**

Wecanuseprint()statementwith Stringandanynumberofarguments.

1)s="Durga"

|  |
| --- |
| **2)a =48** |
| **3)s1="Java"** |
| **4)s2="Python"** |
| **5)print("Hello",s,"YourAge is",a)** |
| **6)print("Youareteaching",s1,"and",s2)** |

**Output:**

Hello DurgaYourAgeis48

You areteachingjavaand Python

**Form-7:print(formattedstring)**

1. **%i** **int**
2. **%d****int**
3. **%f****float**
4. **%s****Stringtype**

Syntax:print("formattedstring"%(variablelist))



Eg 1:

1)a=10

**4)print("avalue is %i"%a)**

**2)b=20**

3)c=30

5)print("bvalueis %dandc valueis%d"%(b,c))

Output

avalueis 10

bvalueis20andcvalueis30 Eg 2:

1)s="Durga"

**2)list=[10,20,30,40]**

3)print("Hello %s...The ListofItemsare%s"%(s,list))

Output:HelloDurga...TheListofItemsare[10,20,30,40]

Form-8:print()withreplacementoperator{} Eg:

1)name = "Durga"

**4)print("Hello{0}yoursalaryis{1}andYourFriend{2}iswaiting". format(name,salary,gf))**

**2)salary = 10000**

3)gf= "Sunny"

5)print("Hello{x}yoursalaryis{y}andYourFriend{z}iswaiting". format(x=name,y=salary,z=gf))

Output

HelloDurgayoursalaryis10000andYourFriendSunnyiswaiting HelloDurgayoursalaryis10000 andYourFriendSunnyiswaiting



**FLOW**

**CONTROL**



Flowcontroldescribestheorderinwhichstatementswillbeexecutedatruntime.

1. **if**
2. **if-elif**
3. **if-elif-else**
4. **break**
5. **continue**
6. **pass**

**Transfer Statements**

1. ConditionalStatements

**ControlFlow**

**Conditional Statements**

1. **for**
2. **while**

**Iterative Statements**

* 1. **if**

ifcondition:statement OR

if condition : statement-1 statement-2 statement-3

Ifconditionistruethenstatementswillbeexecuted. Eg:

* + 1. **name=input("EnterName:")**

**4)print("Howareyou!!!")**

**2)ifname=="durga":**

3) print("HelloDurgaGoodMorning")

D:\Python\_classes>pytest.py Enter Name:durga

HelloDurgaGoodMorning How are you!!!

**57**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



D:\Python\_classes>pytest.py Enter Name: Ravi

How areyou!!!

* 1. **if-else:**

ifcondition:

Action-1 else:

Action-2

ifconditionistruethenAction-1 willbeexecuted otherwiseAction-2will be executed.

* + 1. **name=input("EnterName:")**

|  |
| --- |
| **2)ifname=="durga":** |
| **3) print("HelloDurgaGoodMorning")** |
| **4)else:** |
| **5) print("HelloGuestGood Moring")** |
| **6)print("Howareyou!!!")** |

D:\Python\_classes>pytest.py Enter Name:durga

HelloDurgaGoodMorning How are you!!!

D:\Python\_classes>pytest.py Enter Name:Ravi

HelloGuestGoodMoring How are you!!!

* 1. **if-elif-else:**

ifcondition1:

Action-1 elifcondition2:

Action-2 elifcondition3:

Action-3 elifcondition4:

Action-4

**...**

else:

DefaultAction

Basedconditionthecorrespondingactionwillbe executed.



* + 1. **brand=input("EnterYourFavouriteBrand:")**

3) print("Itischildrensbrand")

5) print("Itisnotthatmuchkick")

7) print("BuyonegetFreeOne")

9) print("OtherBrandsarenotrecommended")

D:\Python\_classes>py test.py EnterYourFavouriteBrand:RC It is childrens brand

**8)else:**

**6)elifbrand=="FO":**

**4)elifbrand=="KF":**

**2)ifbrand=="RC":**

D:\Python\_classes>py test.py EnterYourFavouriteBrand:KF It is not that much kick

D:\Python\_classes>pytest.py

EnterYourFavouriteBrand:KALYANI Other Brands are not recommended

**Note:**

1. **elsepartisalwaysoptional.Hencethefollowingarevariouspossiblesyntaxes.**
   1. **If**
   2. **if–else**
   3. **if-elif-else**
   4. **if-elif**
2. **Thereisnoswitchstatementin Python**
3. **WriteaProgramtofindBiggest ofgiven2NumbersfromtheCommad Prompt?**
   1. **n1=int(input("EnterFirstNumber:"))**

|  |
| --- |
| **2)n2=int(input("Enter SecondNumber:"))** |
| **3)ifn1>n2:** |
| **4) print("BiggestNumberis:",n1)** |
| **5)else:** |
| **6) print("BiggestNumberis:",n2)** |

D:\Python\_classes>pytest.py Enter First Number:10

EnterSecondNumber:20 Biggest Number is: 20



1. **WriteaProgramtofindBiggest ofgiven3NumbersfromtheCommad Prompt?**
   1. **n1=int(input("EnterFirstNumber:"))**

**8)else:**

**6)elifn2>n3:**

**4)ifn1>n2andn1>n3:**

**2)n2=int(input("Enter SecondNumber:"))**

3)n3=int(input("EnterThirdNumber:"))

5) print("BiggestNumberis:",n1)

7) print("BiggestNumberis:",n2)

9) print("BiggestNumberis:",n3)

D:\Python\_classes>pytest.py Enter First Number:10

EnterSecondNumber:20 Enter Third Number:30 Biggest Number is: 30

D:\Python\_classes>pytest.py Enter First Number:10

EnterSecondNumber:30 Enter Third Number:20 Biggest Number is: 30

Q)Writeaprogramtofindsmallestofgiven2 numbers?

Q)Writeaprogramtofindsmallestofgiven3 numbers?

Q)Writeaprogramtocheckwhetherthegivennumberisevenorodd?

**Q)WriteaProgramtoCheckwhetherthegivenNumberisinbetween 1 and 100?**

1)n=int(input("Enter Number:"))

**4)else:**

**2)ifn>=1andn<=10 :**

3) print("Thenumber",n,"isinbetween 1to10")

5) print("Thenumber",n,"isnotinbetween1to10")



**Q)WriteaProgramtotakeaSingleDigitNumberfromtheKeyBoard and Print is Value in English Word?**

1)0ZERO

|  |
| --- |
| **2)1****ONE** |
| **3)** |
| **4)n=int(input("Enteradigitfrom oto9:"))** |
| **5)ifn==0:** |
| **6) print("ZERO")** |
| **7)elifn==1:** |
| **8) print("ONE")** |
| **9)elifn==2:** |
| **10) print("TWO")** |
| **11)elifn==3:** |
| **12) print("THREE")** |
| **13)elifn==4:** |
| **14) print("FOUR")** |
| **15)elifn==5:** |
| **16) print("FIVE")** |
| **17)elifn==6:** |
| **18) print("SIX")** |
| **19)elifn==7:** |
| **20) print("SEVEN")** |
| **21)elifn==8:** |
| **22) print("EIGHT")** |
| **23)elifn==9:** |
| **24) print("NINE")** |
| **25)else:** |
| **26) print("PLEASEENTERA DIGITFROM0TO9")** |



1. IterativeStatements

֍Ifwewanttoexecuteagroupofstatementsmultipletimesthenweshouldgofor Iterative statements.

֍Pythonsupports2typesofiterative statements.

* 1. **forloop**
  2. **whileloop**

1. forloop:

Ifwewanttoexecutesomeactionforeveryelementpresentinsomesequence (it may be string or collection) then we should go for for loop.

**Syntax:forxinsequence:**

Body

Wheresequencecanbestringoranycollection.

Bodywillbeexecutedforeveryelementpresentinthe sequence.

Eg1:Toprintcharacterspresentinthegivenstring

* 1. **s="SunnyLeone"**

**2)forxins:**

3) print(x)

**Output**

S

u n n y

L

e o n e

Eg2:Toprintcharacterspresentinstringindexwise:

1)s=input("EntersomeString: ")

**4) print("Thecharacterpresentat",i,"indexis:",x)**

**2)i=0**

3)forxins:

5) i=i+1



D:\Python\_classes>py test.py EntersomeString:SunnyLeone

The character present at0 index is : S The character present at1 index is : u The character present at2 index is : n The character present at3 index is : n The character present at4 index is : y The character present at5 index is : The character present at6 index is : L The character present at7 index is : e The character present at8 index is : o The character present at9 index is : n Thecharacterpresentat10indexis:e

**Eg3:To printHello10times**

**1)forxin range(10):**

**2) print("Hello")**

Eg4:To displaynumbersfrom0to10

**1)forxin range(11):**

**2) print(x)**

Eg5:To displayoddnumbersfrom0to20

1)forxin range(21):

**2) if(x%2!=0):**

3) print(x)

Eg6:Todisplaynumbersfrom10to 1indescending order

**1)forxin range(10,0,-1):**

**2) print(x)**

Eg7:Toprintsumof numberspresenstinsidelist

**1)list= eval(input("Enter List:"))**

**3) forxinlist:**

**5) print("TheSum=",sum)**

**sum=sum+x;**

**4)**

**2) sum=0;**



D:\Python\_classes>pytest.py Enter List:[10,20,30,40]

TheSum=100

D:\Python\_classes>pytest.py Enter List:[45,67]

TheSum=112

1. whileloop:

Ifwewanttoexecuteagroupofstatementsiterativelyuntilsomeconditionfalse,then we should go for while loop.

**Syntax:whilecondition:**

body

Eg:Toprintnumbersfrom1to10byusingwhileloop

* 1. **x=1**

**4)x=x+1**

**2)while x <= 10:**

3) print(x)

Eg:Todisplaythesumoffirstn numbers

1)n=int(input("Enter number:"))

**6) i=i+1**

**4)whilei<=n:**

**2)sum=0**

3)i=1

5) sum=sum+i

7)print("Thesumoffirst",n,"numbers is:",sum)

Eg:WriteaprogramtopromptusertoentersomenameuntilenteringDurga

1)name=""

**4)print("Thanksforconfirmation")**

**2)whilename!="durga":**

3) name=input("EnterName:")



InfiniteLoops:

1)i=0;

**4) print("Hello",i)**

**2)whileTrue:**

3) i=i+1;

NestedLoops:

Sometimeswecantakealoopinsideanotherloop,whicharealsoknownasnested loops.

1)for iinrange(4):

**2) for jin range(4):**

3) print("i=",i,"j=",j)

**Output**

D:\Python\_classes>pytest.py

|  |  |
| --- | --- |
| **i=0** | **j=0** |
| **i=0** | **j=1** |
| **i=0** | **j=2** |
| **i=0** | **j=3** |
| **i=1** | **j=0** |
| **i=1** | **j=1** |
| **i=1** | **j=2** |
| **i=1** | **j=3** |
| **i=2** | **j=0** |
| **i=2** | **j=1** |
| **i=2** | **j=2** |
| **i=2** | **j=3** |
| **i=3** | **j=0** |
| **i=3** | **j=1** |
| **i=3** | **j=2** |
| **i=3** | **j=3** |

**Q)WriteaProgramtodispaly\*'sinRightAngledTriangledForm**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**1)**

**2)**

**3)**

**4)**

**5)**

**n=int(input("Enternumberofrows:"))**

**for i in range(1,n+1): forjinrange(1,i+1):**

**print("\*",end="")**

**print()**

**AlternativeWay**

**1)n =int(input("Enternumber ofrows:"))**

**3) print("\*"\*i)**

**2)for iinrange(1,n+1):**



**Q)WriteaProgramtodisplay\*'sinPyramidStyle(Also known as EquivalentTriangle)**

**1)n=int(input("Enternumberof rows:"))**

**3) print(""\* (n-i),end="")**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**4) print("\*"\*i)**

**2)for i inrange(1,n+1):**

1. TransferStatements
   1. break:

Wecanusebreakstatementinsideloopstobreakloopexecutionbasedonsome condition.

**1) for iinrange(10):**

**3)**

**print("processingisenough..plzbreak")**

**5) print(i)**

**break**

**4)**

**2) ifi==7:**

D:\Python\_classes>pytest.py 0

1

2

3

4

5

6

processingisenough..plzbreak Eg:

* + 1. **cart=[10,20,600,60,70]**

|  |
| --- |
| **2)foritemin cart:** |
| **3) ifitem>500:** |
| **4) print("Toplacethisorderinsurencemustberequired")** |
| **5) break** |
| **6) print(item)** |



D:\Python\_classes>pytest.py 10

20

Toplacethisorderinsurencemust berequired

* 1. continue:

Wecanusecontinuestatementtoskipcurrentiterationandcontinuenextiteration.

Eg1:Toprintodd numbersin therange0to9

* + 1. **for iinrange(10):**

**4) print(i)**

**2) ifi%2==0:**

3) continue

D:\Python\_classes>pytest.py 1

3

5

7

9

**Eg2:**

1)cart=[10,20,500,700,50,60]

|  |
| --- |
| **2)foritemin cart:** |
| **3) ifitem>=500:** |
| **4) print("Wecannotprocessthisitem :",item)** |
| **5) continue** |
| **6) print(item)** |

D:\Python\_classes>pytest.py 10

20

Wecannotprocessthisitem:500 Wecannotprocessthisitem:700 50

60



**Eg3:**

1)numbers=[10,20,0,5,0,30]

|  |
| --- |
| **2)for ninnumbers:** |
| **3) ifn==0:** |
| **4) print("Heyhowwecandividewithzero..just skipping")** |
| **5) continue** |
| **6) print("100/{}= {}".format(n,100/n))** |

**Output**

100/10=10.0

100/20=5.0

Heyhowwecandividewithzero..justskipping 100/5 = 20.0

Heyhowwecandividewithzero..justskipping 100/30 = 3.3333333333333335

LoopswithelseBlock:

* **Insideloopexecution, ifbreakstatementnotexecuted,thenonlyelsepartwillbe executed.**
* **elsemeansloopwithoutbreak.**

1)cart=[10,20,30,40,50]

|  |
| --- |
| **2)foritemin cart:** |
| **3) ifitem>=500:** |
| **4) print("Wecannotprocessthisorder")** |
| **5) break** |
| **6) print(item)** |
| **7)else:** |
| **8) print("Congrats...allitemsprocessed successfully")** |

**Output**

10

20

30

40

50

Congrats...allitemsprocessedsuccessfully



**Eg:**

1)cart=[10,20,600,30,40,50]

|  |
| --- |
| **2)foritemin cart:** |
| **3) ifitem>=500:** |
| **4) print("Wecannotprocessthisorder")** |
| **5) break** |
| **6) print(item)** |
| **7)else:** |
| **8) print("Congrats...allitemsprocessed successfully")** |

**Output D:\Python\_classes>pytest.py 10**

20

Wecannotprocessthis order

**Q)Whatisthedifferencebetweenforloopandwhileloopin Python?**

֍Wecan useloopstorepeat code execution

֍Repeat codeforeveryitemin sequence forloop

֍Repeat code aslongasconditionistruewhile loop

**Q)Howtoexitfromtheloop?Byusingbreakstatement**

**Q)Howtoskipsomeiterationsinsideloop?Byusingcontinuestatement.**

**Q)Whenelsepartwillbeexecutedwrtloops?Ifloopexecutedwithout break**

* 1. pass statement:
* **pass isakeywordinPython.**
* **Inourprogrammingsyntacticallyifblockisrequiredwhichwon'tdoanythingthenwe can define that empty block with pass keyword.**

pass

|-Itisanemptystatement

|-Itisnull statement

|-Itwon'tdo anything

**Eg:if True:**

SyntaxError:unexpectedEOFwhileparsing if True: pass valid



def m1():

SyntaxError:unexpectedEOFwhileparsing def m1(): pass

UseCaseofpass:

Sometimesintheparentclasswehavetodeclareafunction withemptybodyandchild class responsible to provide proper implementation. Such type of empty body we can define by using pass keyword. (It is something like abstract method in Java)

**Eg:defm1():pass**

* + 1. **foriinrange(100):**

**4) else:pass**

**2) ifi%9==0:**

3) print(i)

D:\Python\_classes>pytest.py 0

9

18

27

36

45

54

63

72

81

90

99

del Statement:

* **delisakeywordinPython.**
* **Afterusingavariable,itis highlyrecommendedtodeletethatvariableifitisnolonger required,so that the corresponding object is eligible for Garbage Collection.**
* **Wecandeletevariablebyusingdel keyword.**

1)x=10

**2)print(x)**

3)delx



AfterdeletingavariablewecannotaccessthatvariableotherwisewewillgetNameError.

1)x=10

**2)delx**

3)print(x)

NameError:name'x'isnotdefined.

Note:Wecandeletevariableswhicharepointingtoimmutableobjects.Butwecannot delete the elements present inside immutable object.

1)s="durga"

**4)del s[0]****TypeError: 'str'object doesn'tsupportitem deletion**

**2)print(s)**

3)delsvalid

DifferencebetweendelandNone:

Inthecasedel,thevariablewillberemovedandwecannotaccessthatvariable(unbind operation)

1)s="durga"

**2) dels**

3) print(s)NameError:name's'isnotdefined.

ButinthecaseofNoneassignmentthevariablewon'tberemovedbutthecorresponding object is eligible for Garbage Collection (re bind operation). Hence after assigning with None value, we can access that variable.

1) s="durga"

**2) s=None**

3) print(s)None



**72**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

# STRING

**DATATYPE**



ThemostcommonlyusedobjectinanyprojectandinanyprogramminglanguageisString only. Hence we should aware complete information about String data type.

What is String?

Anysequenceofcharacterswithineithersinglequotesordoublequotesisconsideredasa String.

**Syntax:**

s = 'durga' s="durga"

Note:In most of other languges like C, C++, Java, a single character with in single quotes istreatedaschardatatypevalue.ButinPythonwearenothavingchardatatype.Henceit is treated as String only.

**Eg:**

>>> ch ='a'

>>>type(ch)

<class'str'>

Howtodefinemulti-lineStringLiterals?

Wecandefinemulti-lineStringliteralsbyusingtriplesingleordoublequotes.

**Eg:**

>>>s='''durgasoftwaresolutions'''

Wecanalsousetriplequotestousesinglequotesordoublequotesassymbolinside String literal.

1. **s = 'Thisis'singlequotesymbol'****Invalid**
2. **s = 'Thisis \'singlequotesymbol'****Valid**
3. **s= "Thisis'singlequotesymbol"****Valid**
4. **s= 'Thisis" doublequotessymbol'****Valid**
5. **s='The"PythonNotes"by'durga'isveryhelpful'****Invalid**
6. **s="The"PythonNotes"by'durga'isveryhelpful"****Invalid**
7. **s= 'The\"PythonNotes\"by\'durga\'isveryhelpful'****Valid**
8. **s='''The"PythonNotes"by'durga'isveryhelpful'''****Valid**



**HowtoAccess Characters ofa String?**

Wecanaccess charactersof astringbyusingthefollowingways.

* 1. **Byusingindex**
  2. **Byusingslice operator**

1. AccessingCharactersByusingIndex:
   * **Pythonsupportsboth +veand-veIndex.**

 +veIndexmeansLefttoRight (Forward Direction)

* + **-veIndexmeansRighttoLeft(BackwardDirection)**

**Eg:s='durga'**

1)>>>s='durga'

**8)>>>s[10]**

**6)>>>s[-1]**

**4)>>>s[4]**

**2)>>>s[0]**

3)'d'

5)'a'

7)'a'

9)IndexError:stringindexoutofrange

Note:Ifwearetryingtoaccesscharactersofastringwithoutofrangeindexthenwewill get error saying: IndexError

**Q)WriteaProgramtoAcceptsome StringfromtheKeyboardanddisplayitsCharacters by Index wise (both Positive and Negative Index)**

**test.py:**

1)s=input("Enter Some String:")

**4) print("Thecharacterpresentatpositiveindex{}andatnEgativeindex{}is{}".fo rmat(i,i-len(s),x))**

**2)i=0**

3)forxin s:

5) i=i+1

Output:D:\python\_classes>pytest.py Enter Some String:durga

Thecharacterpresentatpositiveindex0andatnEgativeindex -5isd Thecharacterpresentatpositiveindex1andatnEgativeindex -4isu Thecharacterpresent atpositiveindex 2 andatnEgativeindex -3isr Thecharacterpresentatpositiveindex3andatnEgativeindex -2isg Thecharacterpresentatpositiveindex4andatnEgativeindex -1isa



1. AccessingCharactersbyusingSliceOperator:
   * **Syntax:s[bEginindex:endindex:step]**
   * **BeginIndex:Fromwherewehavetoconsiderslice(substring)**
   * **EndIndex:Wehavetoterminatetheslice(substring)atendindex-1**
   * **Step:Incremented Value.**

**Note:**

* + **IfwearenotspecifyingbEginindexthenitwillconsiderfrombEginningofthe string.**
  + **Ifwearenotspecifyingendindexthenitwill consideruptoendofthe string.**
  + **Thedefaultvalueforstepis1.**

1)>>>s="LearningPythonisveryvery easy!!!"

**16)>>>s[::-1]**

**14)>>>s[:]**

**12)>>>s[::]**

**10)>>>s[7:]**

**8)>>>s[:7]**

**6)>>>s[1:7:2]**

**4)>>>s[1:7]**

**2)>>>s[1:7:1]**

3)'earnin'

5)'earnin'

7)'eri'

9)'Learnin'

11)'gPythonisveryveryeasy!!!'

13)'LearningPythonisveryveryeasy!!!'

15)'LearningPythonisveryveryeasy!!!'

17)'!!!ysaeyrev yrevsinohtyPgninraeL'

BehaviourofSliceOperator:

1. **s[bEgin:end:step]**
2. **Stepvaluecanbeeither+veor–ve**
3. **If+vethenitshouldbeforwarddirection(lefttoright)andwe havetoconsiderbEgin to end-1**
4. **If-vethenitshouldbebackwarddirection (righttoleft)andwehavetoconsiderbEgin to end+1.**

**\*\*\*Note:**

* + **Inthebackwarddirectionifendvalueis-1thenresultisalwaysempty.**
  + **Intheforwarddirectionifendvalueis0thenresultisalways empty.**



InForwardDirection:

defaultvalueforbEgin:0

defaultvalueforend:lengthofstring default value for step: +1

InBackwardDirection:

defaultvalueforbEgin: -1

defaultvalueforend:-(lengthofstring+1)

Note:Eitherforwardorbackwarddirection,wecantakeboth+veand -vevaluesfor bEgin and end index.

SliceOperatorCaseStudy:

1. **S='abcdefghij'**
2. **s[1:6:2]****'bdf'**
3. **s[::1]****'abcdefghij'**
4. **s[::-1]****'jihgfedcba'**
5. **s[3:7:-1]****''**
6. **s[7:4:-1]****'hgf'**
7. **s[0:10000:1]****'abcdefghij'**
8. **s[-4:1:-1]****'gfedc'**
9. **s[-4:1:-2]****'gec'**
10. **s[5:0:1]****''**
11. **s[9:0:0]****ValueError:slicestepcannotbezero**
12. **s[0:-10:-1]****''**
13. **s[0:-11:-1]****'a'**
14. **s[0:0:1]****''**
15. **s[0:-9:-2]****''**
16. **s[-5:-9:-2]****'fd'**
17. **s[10:-1:-1]****''**
18. **s[10000:2:-1]****'jihgfed'**

Note:SliceoperatorneverraisesIndexError

MathematicalOperatorsforString:

WecanapplythefollowingmathematicaloperatorsforStrings.

1)+ operatorfor concatenation

2)\* operatorfor repetition

* + - **print("durga"+"soft")****durgasoft**
    - **print("durga"\*2)****durgadurga**



**Note:**

1. **Touse+operatorforStrings,compulsorybothargumentsshouldbestrtype.**
2. **Touse\*operatorforStrings,compulsoryoneargumentshouldbestrandother argument should be int.**

len()in-builtFunction:

Wecanuselen()functiontofindthenumberofcharacterspresentinthestring. Eg:

s = 'durga' print(len(s))5

1. **Write a Program to access each Character of String inForwardandBackwardDirectionbyusingwhileLoop?**
   1. **s="LearningPythonisveryeasy!!!"**

|  |
| --- |
| **2)n=len(s)** |
| **3)i=0** |
| **4)print("Forward direction")** |
| **5)whilei<n:** |
| **6) print(s[i],end='')** |
| **7) i+=1** |
| **8)print("Backward direction")** |
| **9)i=-1** |
| **10)whilei>=-n:** |
| **11) print(s[i],end='')** |
| **12) i= i-1** |

**Alternativeways:**

1)s= "Learning Pythonisveryeasy!!!"

|  |
| --- |
| **2)print("Forward direction")** |
| **3)for iins:** |
| 1. **print(i,end='')** 2. **print("Forwarddirection")** |
| **6)for iins[::]:** |
| **7) print(i,end='')** |
| **8)** |
| **9)print("Backward direction")** |
| **10)foriins[::-1]:** |
| **11) print(i,end='')** |



CheckingMembership:

Wecancheckwhetherthecharacterorstringisthememberofanotherstringornotby using in and not in operators

s='durga'

print('d'ins)Trueprint('z'ins)False

1)s =input("Entermain string:")

|  |
| --- |
| **2)subs=input("Entersubstring:")** |
| **3)ifsubsins:** |
| **4) print(subs,"isfoundinmainstring")** |
| **5)else:** |
| **6) print(subs,"isnotfoundinmain string")** |

**Output:**

D:\python\_classes>pytest.py

Entermainstring:durgasoftwaresolutions Enter sub string:durga

durgaisfoundinmain string

D:\python\_classes>pytest.py

Entermainstring:durgasoftwaresolutions Enter sub string:python

pythonisnotfoundinmain string

ComparisonofStrings:

* **Wecanusecomparisonoperators(<,<=,>,>=)andequalityoperators (==,!=)for strings.**
* **Comparisonwillbeperformedbasedonalphabeticalorder.**

1)s1=input("Enterfirst string:")

|  |
| --- |
| **2)s2=input("EnterSecondstring:")** |
| **3)ifs1==s2:** |
| **4) print("Bothstringsareequal")** |
| **5)elifs1<s2:** |
| **6) print("FirstStringislessthanSecondString")** |
| **7)else:** |
| **8) print("FirstStringisgreaterthanSecond String")** |

Output:D:\python\_classes>pytest.py Enter first string:durga



EnterSecondstring:durga Both strings are equal

D:\python\_classes>pytest.py Enter first string:durga

EnterSecond string:ravi

FirstStringislessthanSecond String

D:\python\_classes>pytest.py Enter first string:durga

EnterSecond string:anil

FirstStringisgreaterthanSecondString

RemovingSpacesfromtheString:

Wecan usethefollowing3 methods

1. **rstrip()****Toremovespacesatrighthandside**
2. **lstrip()****Toremovespacesatlefthandside**
3. **strip()****Toremovespacesboth sides**

1)city=input("Enteryour cityName:")

|  |
| --- |
| **2)scity=city.strip()** |
| **3)ifscity=='Hyderabad':** |
| **4) print("HelloHyderbadi..Adab")** |
| **5)elifscity=='Chennai':** |
| **6) print("HelloMadrasi...Vanakkam")** |
| **7)elifscity=="Bangalore":** |
| **8) print("Hello Kannadiga...Shubhodaya")** |
| **9)else:** |
| **10) print("yourenteredcityisinvalid")** |

FindingSubstrings:

Wecan usethe following4 methods

**Forforwarddirection:**

1. **find()**
2. **index()**

**Forbackwarddirection:**

1. **rfind()**
2. **rindex()**



find():

**s.find(substring)**

Returnsindexoffirstoccurrenceofthegivensubstring.Ifitisnotavailablethenwewill get -1.

1)s="LearningPythonisveryeasy"

**4)print(s.find("r"))#3**

**2)print(s.find("Python"))#9**

1. **print(s.find("Java"))#-1**

5)print(s.rfind("r"))#21

Note:Bydefaultfind()methodcansearchtotalstring.Wecanalsospecifythe boundaries to search.

**s.find(substring,bEgin,end)**

Itwill alwayssearchfrombEginindexto end-1 index.

1)s="durgaravipavanshiva"

**4)print(s.find('z',7,15))#-1**

**2)print(s.find('a'))#4**

3)print(s.find('a',7,15))#10

index():

index()methodisexactlysameasfind()method exceptthatifthespecifiedsubstringis not available then we will get ValueError.

1)s=input("Enter mainstring:")

|  |
| --- |
| **2)subs=input("Enter substring:")** |
| **3)try:** |
| **4) n=s.index(subs)** |
| **5)exceptValueError:** |
| **6) print("substringnotfound")** |
| **7)else:** |
| **8) print("substringfound")** |

**Output:**

D:\python\_classes>pytest.py

Entermainstring:learningpythonisveryeasy Enter sub string:python

substringfound



D:\python\_classes>pytest.py

Entermainstring:learningpythonisveryeasy Enter sub string:java

substringnotfound

**Q)ProgramtodisplayallPositionsofSubstringinagivenMain String**

1)s=input("Enter mainstring:")

**12)ifflag==False:**

**10) print("Foundatposition",pos)**

**8) ifpos==-1:**

**6)whileTrue:**

**4)pos=-1**

**2)subs=input("Enter substring:")**

3)flag=False

5)n=len(s)

7) pos=s.find(subs,pos+1,n)

9) break

11) flag=True

13) print("NotFound")

**Output:**

D:\python\_classes>pytest.py

Entermainstring:abbababababacdefg Enter sub string:a

Found at position 0 Found at position 3 Found at position 5 Found at position 7 Found at position 9 Foundatposition11

D:\python\_classes>pytest.py

Entermainstring:abbababababacdefg Enter sub string:bb

Foundatposition1

CountingsubstringinthegivenString:

Wecanfindthenumberofoccurrences ofsubstringpresentinthegivenstringbyusing count() method.

1. **s.count(substring)****Itwillsearchthroughoutthestring.**
2. **s.count(substring,bEgin,end)****ItwillsearchfrombEginindextoend-1 index.**



1)s="abcabcabcabcadda"

3)print(s.count('ab'))

**Output:**

**4)print(s.count('a',3,7))**

**2)print(s.count('a'))**

6

4

2

ReplacingaStringwithanotherString:

**s.replace(oldstring,newstring)**

insides,everyoccurrenceofoldStringwillbereplacedwithnewString.

**Eg1:**

s="LearningPythonisverydifficult"s1 = s.replace("difficult","easy")print(s1)

**Output:LearningPythonisveryeasy**

Eg2:Alloccurrenceswillbereplaced s = "ababababababab"

s1=s.replace("a","b")print(s1)

**Output:bbbbbbbbbbbbbb**

**Q)StringObjectsareImmutablethenhowwecanchangetheContent by using replace() Method**

* **Once we creates string object, we cannot change the content.This non changeable behaviourisnothingbutimmutability.Ifwearetryingtochangethecontentbyusing anymethod,thenwiththosechangesanewobjectwillbecreatedandchangeswon't be happend in existing object.**
* **Hencewithreplace()methodalsoanewobjectgotcreatedbutexistingobjectwon't be changed.**

**Eg:**

s="abab"

s1 = s.replace("a","b")print(s,"is available at :",id(s)) print(s1,"isavailableat:",id(s1))



**Output:**

ababisavailableat:4568672 bbbbisavailableat: 4568704

Intheaboveexample,originalobjectisavailableandwecanseenewobjectwhichwas created because of replace() method.

SplittingofStrings:

* **Wecansplitthegivenstringaccordingtospecifiedseperatorbyusingsplit() method.**
* **l= s.split(seperator)**
* **Thedefaultseperatorisspace.Thereturntypeofsplit()methodis List.**

**1)s="durgasoftware solutions"**

**3)forxin l:**

**print(x)**

**4)**

**2)l=s.split()**

**Output:**

durga software solutions

1)s="22-02-2018"

**4) print(x)**

**2)l=s.split('-')**

3)forxin l:

**Output:**

22

02

2018

JoiningofStrings:

WecanjoinaGroupofStrings(ListORTuple)wrtthegivenSeperator. s=seperator.join(group of strings)

**Eg 1:**

t=('sunny','bunny','chinny')s = '-'.join(t)

print(s)

**Output:sunny-bunny-chinny**



**Eg 2:**

l=['hyderabad','singapore','london','dubai']s = ':'.join(l)

print(s)

Output:hyderabad:singapore:london:dubai

ChangingCaseofaString:

Wecanchangecaseof astringbyusingthefollowing4 methods.

1. **upper()****Toconvert allcharacterstoupper case**
2. **lower()****Toconvertallcharacterstolower case**
3. **swapcase()****Convertsalllowercasecharacterstouppercase andalluppercase characters to lower case**
4. **title()****Toconvertallcharactertotitlecase.i.efirstcharacterineverywordshould be upper case and all remaining characters should be in lower case.**
5. **capitalize()****Onlyfirstcharacterwillbeconverted touppercaseandallremaining characters can be converted to lower case**
   1. **s='learningPythonisvery Easy'**

|  |
| --- |
| **2)print(s.upper())** |
| **3)print(s.lower())** |
| **4)print(s.swapcase())** |
| **5)print(s.title())** |
| **6)print(s.capitalize())** |

**Output:**

LEARNINGPYTHONISVERYEASY

learning python is very easy LEARNINGpYTHONISVERYeASY

LearningPythonIsVeryEasy Learningpythonisvery easy

CheckingStartingandEndingPartoftheString:

Pythoncontainsthefollowingmethodsforthispurpose

* + 1. **s.startswith(substring)**
    2. **s.endswith(substring)**

1)s= 'learningPython isveryeasy'

**4)print(s.endswith('easy'))**

**2)print(s.startswith('learning'))**

3)print(s.endswith('learning'))



**Output:**

**True False True**

ToCheckTypeofCharactersPresentinaString:

Pythoncontainsthefollowingmethodsforthispurpose.

1. **isalnum():ReturnsTrueif allcharactersarealphanumeric(ato z,Ato Z,0to9 )**
2. **isalpha():ReturnsTrueifallcharactersareonlyalphabetsymbols(atoz,AtoZ)**
3. **isdigit(): ReturnsTrueifallcharactersaredigitsonly(0to9)**
4. **islower():ReturnsTrueifallcharactersarelowercasealphabetsymbols**
5. **isupper():ReturnsTrueifallcharactersareuppercaseaplhabet symbols**
6. **istitle(): ReturnsTrueifstringisintitlecase**
7. **isspace():ReturnsTrueifstringcontainsonly spaces**

**Eg:**

1. **print('Durga786'.isalnum())****True**
2. **print('durga786'.isalpha())****False**
3. **print('durga'.isalpha())****True**
4. **print('durga'.isdigit())****False**
5. **print('786786'.isdigit())****True**
6. **print('abc'.islower())****True**
7. **print('Abc'.islower())****False**
8. **print('abc123'.islower())****True**
9. **print('ABC'.isupper())****True**
10. **print('LearningpythonisEasy'.istitle())****False**
11. **print('LearningPythonIsEasy'.istitle())****True**
12. **print(' '.isspace())****True**

**Demo Program:**

* 1. **s=input("Enteranycharacter:")**

|  |
| --- |
| **2)ifs.isalnum():** |
| **3) print("AlphaNumericCharacter")** |
| **4) ifs.isalpha():** |
| **5) print("Alphabetcharacter")** |
| **6) ifs.islower():** |
| **7) print("Lowercasealphabetcharacter")** |
| **8) else:** |
| **9) print("Uppercasealphabetcharacter")** |
| **10) else:** |
| **11) print("itisadigit")** |
| **12)elifs.isspace():** |



1. **print("Itisspace character")**

15) print("NonSpaceSpecialCharacter")

D:\python\_classes>pytest.py Enter any character:7

**14)else:**

AlphaNumericCharacter it is a digit

D:\python\_classes>pytest.py Enter any character:a

AlphaNumericCharacter Alphabet character

Lowercasealphabet character

D:\python\_classes>pytest.py Enter any character:$

NonSpaceSpecial Character

D:\python\_classes>pytest.py Enter any character:A

AlphaNumericCharacter Alphabet character

Uppercasealphabet character

FormattingtheStrings:

Wecanformatthestringswithvariablevaluesbyusingreplacementoperator{}and format() method.

1)name = 'durga'

|  |
| --- |
| **2)salary = 10000** |
| **3)age= 48** |
| **4)print("{}'s salary is{}andhis ageis {}".format(name,salary,age))** |
| **5)print("{0}'s salaryis {1}andhis ageis {2}".format(name,salary,age))** |
| **6)print("{x}'s salaryis {y}and his ageis{z}".format(z=age,y=salary,x=name))** |

**Output:**

**durga'ssalaryis10000andhisageis48 durga'ssalaryis10000andhisageis48 durga'ssalaryis10000andhis ageis48**



ImportantProgramsregardingStringConcept

**Q1)WriteaProgramtoReversethegiven String**

Input: durga Output:agrud

**1stWay:**

**1)s=input("EnterSome String:")**

**2)print(s[::-1])**

**2ndWay:**

**1)s=input("EnterSome String:")**

**2)print(''.join(reversed(s)))**

**3rdWay:**

1)s=input("EnterSome String:")

**6) i=i-1**

**4)whilei>=0:**

**2)i=len(s)-1**

3)target=''

5) target=target+s[i]

7)print(target)

**Q2)ProgramtoReverseOrderofWords**

Input: Learning Python is very Easy Output:EasyVeryisPythonLearning

|  |
| --- |
| **1)s=input("Enter Some String:")** |
| **2)l=s.split()** |
| **3)l1=[]** |
| **4)i=len(l)-1** |
| **5)whilei>=0:** |
| **6) l1.append(l[i])** |
| **7) i=i-1** |
| **8)output=''.join(l1)** |
| **9)print(output)** |

Output:EnterSomeString:LearningPythonisveryeasy!! easy!!! very is Python Learning



**Q3)ProgramtoReverseInternalContentofeach Word**

Input: Durga Software Solutions Output:agruDerawtfoSsnoituloS

1) s=input("EnterSomeString:")

**8)output=''.join(l1)**

**6) l1.append(l[i][::-1])**

**4)i=0**

**2)l=s.split()**

3)l1=[]

5)whilei<len(l):

7) i=i+1

9)print(output)

**Q4)WriteaProgramtoPrintCharactersatOddPositionandEven Position for the given String?**

**1stWay:**

s = input("Enter Some String:")print("CharactersatEvenPosition:",s[0::2])print("Characters at Odd Position:",s[1::2])

**2ndWay:**

1)s=input("Enter Some String:")

|  |
| --- |
| **2)i=0** |
| **3)print("Characters atEven Position:")** |
| **4)while i< len(s):** |
| **5) print(s[i],end=',')** |
| **6) i=i+2** |
| **7)print()** |
| **8)print("Characters atOdd Position:")** |
| **9)i=1** |
| **10)whilei<len(s):** |
| **11) print(s[i],end=',')** |
| **12) i=i+2** |



**Q5)ProgramtoMergeCharactersof2StringsintoaSingleString by taking Characters alternatively**

Input:s1= "ravi"

s2="reja" Output: rtaevjia

1)s1=input("EnterFirstString:")

|  |
| --- |
| **2)s2=input("EnterSecondString:")** |
| **3)output=''** |
| **4)i,j=0,0** |
| **5)whilei<len(s1)orj<len(s2):** |
| **6) ifi<len(s1):** |
| **7) output=output+s1[i]** |
| **8) i+=1** |
| **9) ifj<len(s2):** |
| **10) output=output+s2[j]** |
| **11) j+=1** |
| **12)print(output)** |

**Output:**

Enter First String:durga EnterSecondString:ravisoft druarvgiasoft

**Q6)WriteaProgramtoSorttheCharactersoftheStringandFirst Alphabet Symbols followed by Numeric Values**

Input: B4A1D3 Output:ABD134

1)s=input("Enter Some String:")

|  |
| --- |
| **2)s1=s2=output=''** |
| **3)forxin s:** |
| **4) ifx.isalpha():** |
| **5) s1=s1+x** |
| **6) else:** |
| **7) s2=s2+x** |
| **8)forxin sorted(s1):** |
| **9) output=output+x** |
| **10)for xinsorted(s2):** |
| **11) output=output+x** |
| **12)print(output)** |



**Q7)WriteaProgramforthefollowing Requirement**

Input: a4b3c2 Output:aaaabbbcc

**1)s=input("Enter Some String:")**

**3)forxin s:**

**5)**

**output=output+x**

**7) else:**

**9)print(output)**

**output=output+previous\*(int(x)-1)**

**8)**

**previous=x**

**6)**

**4) ifx.isalpha():**

**2)output=''**

Note: chr(unicode) The corresponding character ord(character)Thecorrespondingunicodevalue

**Q8)WriteaProgramtoperformthefollowingActivity**

Input: a4k3b2 Outpt:aeknbd

**1)s=input("Enter Some String:")**

**3)forxin s:**

**5)**

**output=output+x**

**7) else:**

**9)print(output)**

**output=output+chr(ord(previous)+int(x))**

**8)**

**previous=x**

**6)**

**4) ifx.isalpha():**

**2)output=''**

**Q9)WriteaProgramtoRemoveDuplicateCharactersfromthe given Input String?**

Input:ABCDABBCDABBBCCCDDEEEF

Output:ABCDEF

1)s=input("EnterSome String:")

**6)output=''.join(l)**

**4) ifxnot in l:**

**2)l=[]**

3)forxin s:

5) l.append(x)

**7)print(output)**



**Q10)WriteaProgramtofindtheNumberofOccurrencesofeach Character present in the given String?**

**Input:ABCABCABBCDE**

**Output:A-3,B-4,C-3,D-1,E-1**

1)s=input("EntertheSome String:")

**8)fork,vind.items():**

**6) else:**

**4) ifxin d.keys():**

**2)d={}**

3)forxin s:

5) d[x]=d[x]+1

7) d[x]=1

9) print("{}={}Times".format(k,v))

**Q11)WriteaProgramtoperformthefollowingTask?**

Input: 'one two three four five six seven' Output:'oneowtthreeruoffivexisseven'

1)s=input('EnterSomeString:')

**12)print('OriginalString:',s)**

**10) i=i+1**

**8) else:**

**6) ifi%2==0:**

**4)i=0**

**2)l=s.split()**

3)l1=[]

5)whilei<len(l):

7) l1.append(l[i])

9) l1.append(l[i][::-1])

11)output=''.join(l1)

13)print('outputString:',output)

**Output:**

D:\durgaclasses>pytest.py

EnterSomeString:onetwothreefourfivesixseven Original String: one two three four five six seven output String: one owt three ruof five xis seven



FormattingtheStrings:

֍Wecanformatthestringswithvariablevaluesbyusingreplacementoperator{}and format() method.

֍Themainobjectiveofformat()methodtoformatstringintomeaningfuloutput form.

Case-1:Basicformattingfordefault,positionalandkeywordarguments

1)name = 'durga'

|  |
| --- |
| **2)salary = 10000** |
| **3)age= 48** |
| **4)print("{}'s salary is{}andhis ageis {}".format(name,salary,age))** |
| **5)print("{0}'s salaryis {1}andhis ageis {2}".format(name,salary,age))** |
| **6)print("{x}'s salaryis {y}and his ageis{z}".format(z=age,y=salary,x=name))** |

**Output:**

durga'ssalaryis10000andhisageis48 durga'ssalaryis10000andhisageis48 durga'ssalaryis10000and hisageis48

**Case-2:FormattingNumbers d** **Decimal IntEger**

fFixedpointnumber(float).Thedefaultprecisionis6 b Binary format

oOctalFormat

xHexaDecimalFormat(Lowercase) XHexaDecimalFormat (Uppercase)

**Eg-1:**

1. **print("TheintEgernumberis:{}".format(123))**

**4)print("TheintEger number is: {:05d}".format(123))**

**2)print("TheintEger number is: {:d}".format(123))**

3)print("TheintEger number is: {:5d}".format(123))

**Output:**

The intEger number is: 123 The intEger number is: 123 The intEger number is:123 TheintEgernumberis:00123

**92**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

**Eg-2:**

1)print("Thefloatnumberis: {}".format(123.4567))

|  |
| --- |
| **2)print("Thefloatnumberis: {:f}".format(123.4567))** |
| **3)print("Thefloatnumberis: {:8.3f}".format(123.4567))** |
| **4)print("Thefloatnumberis: {:08.3f}".format(123.4567))** |
| **5)print("Thefloatnumberis: {:08.3f}".format(123.45))** |
| **6)print("Thefloatnumberis: {:08.3f}".format(786786123.45))** |

**Output:**

The float number is: 123.4567 Thefloatnumberis:123.456700 The float number is:123.457 The float number is: 0123.457 The float number is: 0123.450

Thefloatnumber is: 786786123.450

**Note:**

֍{:08.3f}

֍Totalpositionsshouldbeminimum8.

֍Afterdecimalpointexactly3digitsareallowed.Ifitislessthen0swillbeplacedinthe last positions

֍If totalnumberis< 8positionsthen0willbeplaced inMSBs

֍Iftotalnumberis>8positionsthenallintEgral digitswill beconsidered.

֍Theextradigits wecan takeonly0

Note:FornumbersdefaultalignmentisRightAlignment(>)

Eg-3:PrintDecimalvalueinbinary,octalandhexadecimalform

1)print("Binary Form:{0:b}".format(153))

**4)print("HexadecimalForm:{0:X}".format(154))**

**2)print("Octal Form:{0:o}".format(153))**

3)print("HexadecimalForm:{0:x}".format(154))

**Output:**

BinaryForm:10011001 Octal Form:231

HexadecimalForm:9a HexadecimalForm:9A

Note:Wecanrepresentonlyintvaluesinbinary,octalandhexadecimalanditisnot possible for float values.



**Note:**

1. **{:5d}IttakesanintEgerargumentandassignsaminimumwidthof5.**
2. **{:8.3f}Ittakesafloatargumentandassignsaminimumwidthof8 including"."and after decimal point excatly 3 digits are allowed with round operation if required**
3. **{:05d}Theblankplacescanbefilledwith0.Inthisplaceonly0 allowed.**

Case-3:Numberformattingforsigned numbers

֍Whiledisplayingpositivenumbers,ifwewant toinclude+ thenwehaveto write

{:+d} and {:+f}

֍Usingplusfor-venumbersthereisnouseandfor -venumbers-signwillcome automatically.

* 1. **print("intvaluewith sign:{:+d}".format(123))**

**4)print("floatvaluewithsign:{:+f}".format(-123.456))**

**2)print("intvaluewithsign:{:+d}".format(-123))**

3)print("float valuewith sign:{:+f}".format(123.456))

**Output:**

intvaluewithsign:+123 int value with sign:-123

floatvaluewithsign:+123.456000 float value with sign:-123.456000

Case-4:Numberformattingwith alignment

֍<,>,^ and= areusedforalignment

֍<LeftAlignmenttotheremainingspace

֍^Centeralignmentto theremainingspace

֍>Rightalignmenttotheremainingspace

֍= Forces the signed(+) (-) to the left most position Note:DefaultAlignmentfornumbersisRightAlignment. Ex:

1)print("{:5d}".format(12))

**8)print("{:^10.3f}".format(12.23456))**

**6)print("{:^5d}".format(12))**

**4)print("{:>5d}".format(12))**

**2)print("{:<5d}".format(12))**

3)print("{:<05d}".format(12))

5)print("{:>05d}".format(12))

7)print("{:=5d}".format(-12))

9)print("{:=8.3f}".format(-12.23456))



**Output:**

12

12

12000

12

00012

12

-12

12.235

-12.235

**Case-5:Stringformattingwithformat()**

Similartonumbers,wecanformatStringvaluesalsowithformat() method. s.format(string)

1)print("{:5d}".format(12))

|  |
| --- |
| **2)print("{:5}".format("rat"))** |
| **3)print("{:>5}".format("rat"))** |
| **4)print("{:<5}".format("rat"))** |
| **5)print("{:^5}".format("rat"))** |
| **6)print("{:\*^5}".format("rat"))#Insteadof\*wecanuseanycharacter(like+,$,aetc)** |

**Output:**

12

rat rat rat rat

\*rat\*

Note:Fornumbersdefaultalignmentisrightwhereasforstringsdefaultalignmentis left

Case-6:TruncatingStringswithformat()method

1)print("{:.3}".format("durgasoftware"))

**4)print("{:^5.3}".format("durgasoftware"))**

**2)print("{:5.3}".format("durgasoftware"))**

3)print("{:>5.3}".format("durgasoftware"))

5)print("{:\*^5.3}".format("durgasoftware"))

**Output:**

dur dur dur



dur

\*dur\*

Case-7:Formattingdictionarymembersusingformat()

**1)person={'age':48,'name':'durga'}**

**2)print("{p[name]}'sageis: {p[age]}".format(p=person))**

**Output:**

durga'sageis: 48

Note: pisaliasnameofdictionary

persondictionarywearepassingaskeywordargument More convinient way is to use \*\*person

**1)person={'age':48,'name':'durga'}**

**2)print("{name}'s ageis: {age}".format(\*\*person))**

**Output:durga'sageis:48**

Case-8:Formattingclassmembersusingformat()

1)classPerson:

**4)print("{p.name}'sageis :{p.age}".format(p=Person()))**

**2) age=48**

3) name="durga"

**Output:durga'sageis:48**

1)classPerson:

|  |
| --- |
| **2) definit(self,name,age):** |
| **3) self.name=name** |
| **4) self.age=age** |
| **5)print("{p.name}'sageis :{p.age}".format(p=Person('durga',48)))** |
| **6)print("{p.name}'sageis :{p.age}".format(p=Person('Ravi',50)))** |

Note:HerePersonobjectispassedaskeywordargument.Wecanaccessbyusingits reference variable in the template string

Case-9:DynamicFormattingusingformat()

1)string="{:{fill}{align}{width}}"

**2)print(string.format('cat',fill='\*',align='^',width=5))**

3)print(string.format('cat',fill='\*',align='^',width=6))



5)print(string.format('cat',fill='\*',align='>',width=6))

**4)print(string.format('cat',fill='\*',align='<',width=6))**

**Output:**

\*cat\*

\*cat\*\* cat\*\*\*

\*\*\*cat

**Case-10:DynamicFloatformat template**

1)num="{:{align}{width}.{precision}f}"

**2)print(num.format(123.236,align='<',width=8,precision=2))**

3)print(num.format(123.236,align='>',width=8,precision=2))

**Output:**

123.24

123.24

**Case-11:FormattingDatevalues**

1)importdatetime

**4)print("It'snow:{:%d/%m/%Y%H:%M:%S}".format(date))**

**2)#datetimeformatting**

3)date=datetime.datetime.now()

**Output:It'snow:09/03/201812:36:26**

**Case-12:Formattingcomplexnumbers**

**1)complexNumber=1+2j**

**2)print("RealPart:{0.real}andImaginary Part:{0.imag}".format(complexNumber))**

Output:RealPart:1.0andImaginaryPart:2.0



#### DATASTRUCTURE

LIST



֍Ifwewanttorepresentagroupofindividualobjectsasasingle entitywhereinsertion order preserved and duplicates are allowed, then we should go for List.

֍insertionorder preserved.

֍duplicateobjectsare allowed.

֍heterogeneousobjectsare allowed.

֍Listisdynamicbecausebasedonourrequirementwecanincreasethe sizeand decrease the size.

֍InListtheelementswillbeplacedwithinsquarebracketsandwithcomma seperator.

֍Wecandifferentiateduplicateelementsbyusingindexandwecanpreserveinsertion order by using index. Hence index will play very important role.

֍Pythonsupportsbothpositiveandnegativeindexes.+veindexmeansfromleftto right where as negative index means right to left.

[10,"A","B",20,30, 10]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **-6** | **-5** | **-4** | **-3** | **-2** | **-1** |
| **10** | **A** | **B** | **20** | **30** | **10** |
| **0** | **1** | **2** | **3** | **4** | **5** |

֍List objectsaremutable.i.ewecanchangethecontent.

CreationofListObjects:

1. **Wecancreateemptylistobjectasfollows...**
   1. **list=[]**

|  |
| --- |
| **2)print(list)** |
| **3)print(type(list))** |
| **4)** |
| **5)[]** |
| **6)<class'list'>** |

1. **Ifweknowelements alreadythenwecancreatelist asfollowslist =[10,20,30,40]**
2. **WithDynamicInput:**
   1. **list=eval(input("EnterList:"))**

**2)print(list)**

3)print(type(list))

D:\Python\_classes>pytest.py Enter List:[10,20,30,40]

[10,20,30,40]

<class'list'>



1. **Withlist()Function:**
   1. **l=list(range(0,10,2))**

**2)print(l)**

3)print(type(l))

D:\Python\_classes>pytest.py [0, 2, 4, 6, 8]

<class'list'>Eg:

1)s="durga"

**2)l=list(s)**

3)print(l)

D:\Python\_classes>pytest.py ['d', 'u', 'r', 'g', 'a']

1. **Withsplit()Function:**
   1. **s="LearningPythonisveryveryeasy!!!"**

**4)print(type(l))**

**2)l=s.split()**

3)print(l)

D:\Python\_classes>pytest.py

['Learning','Python','is','very','very','easy','!!!']

<class'list'>

Note:Sometimeswecantakelistinsideanotherlist, suchtypeoflistsarecallednested lists.

[10,20, [30,40]]

AccessingElementsofList:

Wecanaccesselementsofthelisteitherbyusingindexorbyusingslice operator(:)

1. Byusing Index:

֍Listfollowszerobasedindex. ieindexoffirstelementiszero.

֍Listsupportsboth +veand-veindexes.

֍+veindexmeant forLeftto Right

֍-veindex meantforRight toLeft

֍list=[10, 20,30,40]



list

|  |  |  |  |
| --- | --- | --- | --- |
| **-4** | **-3** | **-2** | **-1** |
| **10** | **20** | **30** | **40** |
| **0** | **1** | **2** | **3** |

֍print(list[0])10

֍print(list[-1])40

֍print(list[10])IndexError:listindexoutofrange

1. ByusingSliceOperator:

**Syntax:list2= list1[start:stop:step]**

StartItindicatestheIndexwhereslicehastoStart Default Value is 0

StopItindicatestheIndexwhereslicehastoEnd

DefaultValueismaxallowedIndexofListieLength oftheList

Stepincrementvalue

DefaultValueis1

**1)n=[1,2,3,4,5,6,7,8,9,10]**

|  |  |
| --- | --- |
| **2)** | **print(n[2:7:2])** |
| **3)print(n[4::2])** | |
| **4)** | **print(n[3:7])** |
| **5)** | **print(n[8:2:-2])** |
| **6)** | **print(n[4:100])** |

Output D:\Python\_classes>pytest.py [3, 5, 7]

[5,7,9]

[4, 5,6,7]

[9,7,5]

[5, 6,7,8,9,10]



ListvsMutability:

OncewecreatesaListobject, wecanmodifyitscontent. HenceList objectsare mutable.

1)n=[10,20,30,40]

**4)print(n)**

**2)print(n)**

3)n[1]=777

D:\Python\_classes>pytest.py [10, 20, 30, 40]

[10,777,30,40]

TraversingtheElementsofList:

Thesequentialaccess ofeach elementinthelistiscalled traversal.

1. ByusingwhileLoop:
   1. **n=[0,1,2,3,4,5,6,7,8,9,10]**

**4) print(n[i])**

**2)i=0**

3)while I<len(n):

5) i=i+1

D:\Python\_classes>pytest.py 0

1

2

3

4

5

6

7

8

9

10

1. Byusingfor Loop:
   1. **n=[0,1,2,3,4,5,6,7,8,9,10]**

**2)for n1inn:**

3) print(n1)



D:\Python\_classes>pytest.py 0

1

2

3

4

5

6

7

8

9

10

1. TodisplayonlyEvenNumbers:

**1)n=[0,1,2,3,4,5,6,7,8,9,10]**

**3) ifn1%2==0:**

**print(n1)**

**4)**

**2)for n1inn:**

D:\Python\_classes>pytest.py 0

2

4

6

8

10

1. TodisplayElementsbyIndexwise:
   1. **l=["A","B","C"]**

**4) print(l[i],"isavailableatpositiveindex:",i,"andatnegativeindex:",i-x)**

**2)x=len(l)**

3)for iinrange(x):

D:\Python\_classes>pytest.py

Aisavailableatpositiveindex:0andat negativeindex:-3

Bis availableat positiveindex:1and atnegativeindex:-2

Cisavailableat positiveindex:2and atnegativeindex:-1



ImportantFunctionsofList:

1. TogetInformationaboutList:
2. **len():**

Returnsthenumberofelementspresentinthelist

Eg:n= [10,20,30,40]

print(len(n)4

1. **count():**

Itreturnsthenumberofoccurrencesofspecifiediteminthelist

* 1. **n=[1,2,2,2,2,3,3]**

**4)print(n.count(3))**

**2)print(n.count(1))**

3)print(n.count(2))

5)print(n.count(4))

D:\Python\_classes>pytest.py 1

4

2

0

1. **index():**

Returnstheindexoffirstoccurrenceofthespecifieditem.

* 1. **n =[1,2,2,2,2,3,3]**

**4)print(n.index(3))****5**

**2)print(n.index(1))****0**

3)print(n.index(2))1

5)print(n.index(4))ValueError:4is notinlist

Note:If the specified element not present in the list then we will get ValueError.Hence beforeindex()methodwehavetocheckwhetheritempresentinthelistornotbyusingin operator.

print( 4in n)False



1. ManipulatingElementsofList:
   1. **append()Function:**

Wecanuseappend()function toadditemat theendofthelist.

* + 1. **list=[]**

**4)list.append("C")**

**2)list.append("A")**

3)list.append("B")

5)print(list)

D:\Python\_classes>pytest.py ['A', 'B', 'C']

Eg:Toaddallelementstolistupto100whicharedivisibleby10

**1)list=[]**

**3) ifi%10==0:**

**5)print(list)**

**list.append(i)**

**4)**

**2)for iinrange(101):**

D:\Python\_classes>pytest.py

[0, 10,20, 30, 40,50,60,70,80,90,100]

* 1. **insert() Function:**

Toinsertitematspecifiedindex position

* + 1. **n=[1,2,3,4,5]**

**2)n.insert(1,888)**

3)print(n)

D:\Python\_classes>pytest.py [1, 888, 2, 3, 4, 5]

1)n=[1,2,3,4,5]

**4)print(n)**

**2)n.insert(10,777)**

3)n.insert(-10,999)

D:\Python\_classes>pytest.py [999, 1, 2, 3, 4, 5, 777]



Note:Ifthespecifiedindexisgreaterthanmaxindexthenelementwillbeinsertedatlast position. If the specified index is smaller than min index then element will be inserted at first position.

**Differencesbetweenappend()andinsert()**

|  |  |
| --- | --- |
| **append()** | **insert()** |
| **InListwhenweaddanyelementitwill come in last i.e. it will be last element.** | **InListwecaninsertanyelementin particular index number** |

* 1. **extend()Function:**

Toaddallitemsofonelistto anotherlist l1.extend(l2)

allitemspresentinl2willbeaddedto l1

* + 1. **order1=["Chicken","Mutton","Fish"]**

**4)print(order1)**

**2)order2=["RC","KF","FO"]**

3)order1.extend(order2)

D:\Python\_classes>pytest.py

['Chicken','Mutton','Fish','RC','KF','FO']

1)order = ["Chicken","Mutton","Fish"]

**2)order.extend("Mushroom")**

3)print(order)

D:\Python\_classes>pytest.py

['Chicken','Mutton','Fish','M','u','s','h','r','o','o','m']

* 1. **remove() Function:**

Wecanusethisfunctiontoremove specifieditemfromthelist.Iftheitempresent multiple times then only first occurrence will be removed.

* + 1. **n=[10,20,10,30]**

**2)n.remove(10)**

3)print(n)

D:\Python\_classes>pytest.py [20, 10, 30]

Ifthespecifieditemnotpresentinlistthenwewillget ValueError



1)n=[10,20,10,30]

3)print(n)

ValueError:list.remove(x):xnotinlist

**2)n.remove(40)**

Note:Hencebeforeusingremove()methodfirstwehavetocheckspecifiedelement present in the list or not by using in operator.

* 1. **pop() Function:**
* **Itremovesandreturnsthelastelementofthelist.**
* **Thisisonlyfunctionwhichmanipulateslistandreturnssome element.**
  + 1. **n=[10,20,30,40]**

**4)print(n)**

**2)print(n.pop())**

3)print(n.pop())

D:\Python\_classes>pytest.py 40

30

[10,20]

Ifthelistis emptythenpop()functionraisesIndexError

**1)n=[]**

**2)print(n.pop())****IndexError:popfromemptylist**

**Note:**

1. **pop()istheonlyfunctionwhichmanipulatesthelistandreturnssomevalue**
2. **Ingeneralwecanuseappend()andpop()functionstoimplementstack datastructure by using list,which follows LIFO(Last In First Out) order.**

Ingeneralwecanusepop()functiontoremovelastelementofthelist.Butwecanuseto remove elements based on index.

n.pop(index)Toremoveandreturnelementpresentatspecifiedindex. n.pop() To remove and return last element of the list

1) n=[10,20,30,40,50,60]

**4) print(n.pop(10))****IndexError:pop indexoutofrange**

**2) print(n.pop())****60**

3) print(n.pop(1))20



**Differencesbetweenremove()andpop()**

|  |  |
| --- | --- |
| **remove()** | **pop()** |
| **1)Wecanusetoremovespecialelement from the List.** | **1)Wecanusetoremovelastelement from the List.** |
| **2) Itcan’treturnanyvalue.** | **2)Itreturnedremovedelement.** |
| **3)Ifspecialelementnotavailablethenwe**  **getVALUE ERROR.** | **3) IfListisemptythenwegetError.** |

Note:ListObjectsaredynamic.i.ebasedonourrequirementwecanincreaseand decrease the size.

append(),insert(),extend()forincreasingthesize/growablenature remove(), pop() for decreasing the size /shrinking nature

1. OrderingElementsofList:
   1. **reverse():**

Wecanusetoreverse()orderofelementsof list.

* + 1. **n=[10,20,30,40]**

**2)n.reverse()**

3)print(n)

D:\Python\_classes>pytest.py [40, 30, 20, 10]

* 1. **sort():**

Inlistbydefaultinsertionorderispreserved.Ifwanttosorttheelementsoflist according to default natural sorting order thenwe should go for sort()method.

* + 1. **Fornumbers****DefaultNatural sortingOrderisAscending Order**
    2. **ForStrings****DefaultNaturalsortingorderisAlphabeticalOrder**

**1)n=[20,5,15,10,0]**

**3)print(n)****[0,5,10,15,20]**

**5)s=["Dog","Banana","Cat","Apple"]**

**7)print(s)****['Apple','Banana','Cat','Dog']**

**6)s.sort()**

**4)**

**2)n.sort()**



Note:Tousesort()function,compulsorylistshouldcontainonlyhomogeneouselements. Otherwise we will get TypeError

1)n=[20,10,"A","B"]

**2)n.sort()**

3)print(n)

TypeError:'<'notsupportedbetweeninstancesof'str'and'int'

Note:InPython2ifListcontainsbothnumbersandStringsthensort()functionfirstsort numbers followed by strings

1)n=[20,"B",10,"A"]

**2)n.sort()**

3)print(n)# [10,20,'A','B']

Butin Python 3itisinvalid.

ToSortinReverseofDefaultNaturalSortingOrder:

Wecansortaccordingtoreverseofdefaultnaturalsortingorderbyusingreverse=True argument.

1)n=[40,10,30,20]

**6)n.sort(reverse =False)**

**4)n.sort(reverse =True)**

**2)n.sort()**

3)print(n)[10,20,30,40]

5)print(n)[40,30,20,10]

7)print(n)[10,20,30,40]

AliasingandCloningofList Objects:

Theprocessofgivinganotherreferencevariabletotheexistinglistiscalledaliasing.

**1)**

**2)**

**3)**

**x=[10,20,30,40]**

**y=x print(id(x))**

**x**

**y**

**4)print(id(y))**

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20** | **30** | **40** |

Theprobleminthisapproachisbyusingonereferencevariableifwearechanging content, then those changes will be reflected to the other reference variable.

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20**  **777** | **30** | **40** |



**4)print(x)****[10,777,30,40]**

Toovercomethisproblemweshouldgoforcloning.

Theprocessofcreatingexactlyduplicateindependentobjectiscalledcloning. Wecanimplementcloningbyusingsliceoperator orbyusingcopy()function.

1. **ByusingSlice Operator:**
   1. **x=[10,20,30,40]**

**4)print(x)****[10, 20,30,40]**

**2)y =x[:]**

3)y[1]= 777

5)print(y)[10,777,30, 40]

x

**y**

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20** | **30** | **40** |

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20**  **777** | **30** | **40** |

1. **Byusingcopy() Function:**
   1. **x=[10,20,30,40]**

**4)print(x)****[10, 20,30,40]**

**2)y =x.copy()**

3)y[1]= 777

5)print(y)[10,777,30, 40]

x

**y**

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20** | **30** | **40** |

|  |  |  |  |
| --- | --- | --- | --- |
| **10** | **20**  **777** | **30** | **40** |

**Q)Differencebetween=Operatorandcopy() Function**

֍=Operatormeantfor aliasing

֍copy()Functionmeantforcloning



UsingMathematicalOperatorsforListObjects:

Wecan use+ and\* operatorsforList objects.

1. **ConcatenationOperator(+):**

Wecanuse+toconcatenate2 listsintoasinglelist

* 1. **a= [10,20,30]**

**4)print(c)****[10,20,30,40, 50, 60]**

**2)b = [40, 50,60]**

3)c=a+b

Note:Touse+operatorcompulsorybothargumentsshouldbelistobjects, otherwisewe will get TypeError.

**Eg:**

c=a+40TypeError:canonlyconcatenatelist(not"int")to list. c = a+[40] Valid

1. **RepetitionOperator(\*):**

Wecanuserepetitionoperator\*torepeatelementsoflistspecifiednumberoftimes.

* 1. **x =[10,20, 30]**

**2)y =x\*3**

3)print(y)[10,20, 30,10, 20, 30,10,20,30]

ComparingListObjects

WecanusecomparisonoperatorsforList objects.

1. **x= ["Dog","Cat", "Rat"]**

|  |
| --- |
| **2)y=["Dog", "Cat","Rat"]** |
| **3)z =["DOG","CAT","RAT"]** |
| **4)print(x== y)****True** |
| **5)print(x ==z)****False** |
| **6)print(x!=z)****True** |

Note:Wheneverweareusingcomparisonoperators (==,!=)forListobjectsthenthe following should be considered

* 1. **TheNumberof Elements**
  2. **TheOrderof Elements**
  3. **TheContentofElements(CaseSensitive)**

Note:WheneverweareusingrelatationalOperators(<,<=,>,>=)betweenListObjects, only 1ST Element comparison will be performed.



1)x =[50,20, 30]

|  |
| --- |
| **2)y=[40,50,60,100,200]** |
| **3)print(x>y)****True** |
| **4)print(x>=y)****True** |
| **5)print(x<y)****False** |
| **6)print(x<=y)****False** |

Eg:

1)x = ["Dog", "Cat", "Rat"]

|  |
| --- |
| **2)y=["Rat", "Cat", "Dog"]** |
| **3)print(x>y)****False** |
| **4)print(x>=y)****False** |
| **5)print(x<y)****True** |
| **6)print(x<=y)****True** |

MembershipOperators:

Wecancheckwhetherelementisamemberofthelistornotbyusingmemebership operators.

1. **inOperator**
2. **notinOperator**

1)n=[10,20,30,40]

**4)print(50inn)**

**2)print(10inn)**

1. **print(10notinn)**

5)print(50 notinn)

**OutputTrue False False True**

clear()Function:

Wecanuseclear()functiontoremoveallelementsof List.

1)n=[10,20,30,40]

**4)print(n)**

**2)print(n)**

3)n.clear()



Output D:\Python\_classes>pytest.py [10, 20, 30, 40]

**[]**

NestedLists:

Sometimeswecantakeonelistinsideanotherlist.Suchtypeoflistsarecallednested lists.

1)n=[10,20,[30,40]]

|  |
| --- |
| **2)print(n)** |
| **3)print(n[0])** |
| **4)print(n[2])** |
| **5)print(n[2][0])** |
| **6)print(n[2][1])** |

Output D:\Python\_classes>pytest.py [10, 20, [30, 40]]

10

[30,40]

30

40

Note:Wecanaccessnestedlistelementsbyusingindexjustlikeaccessingmulti dimensional array elements.

NestedListasMatrix:

InPythonwecanrepresentmatrixbyusingnestedlists.

1)n=[[10,20,30],[40,50,60],[70,80,90]]

|  |
| --- |
| **2)print(n)** |
| **3)print("Elements byRowwise:")** |
| **4)for rinn:** |
| **5) print(r)** |
| **6)print("Elements byMatrixstyle:")** |
| **7)for iinrange(len(n)):** |
| **8) for jin range(len(n[i])):** |
| **9) print(n[i][j],end='')** |
| **10) print()** |



**Output**

D:\Python\_classes>pytest.py

[[10,20,30],[40,50,60],[70,80,90]]

Elements byRowwise:

[10,20,30]

[40,50,60]

[70,80,90]

ElementsbyMatrix style:

10 20 30

40 50 60

70 80 90

ListComprehensions:

Itisveryeasyandcompactwayofcreatinglistobjectsfromanyiterableobjects (Like List, Tuple, Dictionary, Range etc) based on some condition.

Syntax:list =[expressionforiteminlistif condition]

1)s= [x\*xforxinrange(1,11)]

|  |
| --- |
| **2)print(s)** |
| **3)v =[2\*\*xforxinrange(1,6)]** |
| **4)print(v)** |
| **5)m=[xforxinsifx%2==0]** |
| **6)print(m)** |

D:\Python\_classes>pytest.py

[1,4,9,16,25,36, 49,64, 81, 100]

[2, 4,8,16,32]

[4,16,36,64, 100]

1)words=["Balaiah","Nag","Venkatesh","Chiranjeevi"]

**2)l=[w[0]forw inwords]**

3)print(l)

**Output:['B','N','V','C']**

1)num1=[10,20,30,40]

**114**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

|  |
| --- |
| **2)num2=[30,40,50,60]** |
| **3)num3=[iforiin num1 ifinotinnum2]** |
| **4)print(num3)[10,20]** |
| **5)** |
| **6)commonelementspresent innum1andnum2** |



**7)num4=[i for iinnum1ifiin num2]**

**8)print(num4)[30, 40]**

**Eg:**

1)words="thequickbrown foxjumps overthelazy dog".split()

**4)print(l)**

**2)print(words)**

3)l=[[w.upper(),len(w)] forwinwords]

**Output**

['the','quick','brown','fox','jumps','over','the','lazy','dog']

[['THE',3],['QUICK',5],['BROWN',5],['FOX',3],['JUMPS',5],['OVER',4],

['THE',3],['LAZY',4],['DOG',3]]

**Q)WriteaProgramtodisplayUniqueVowelspresentinthegiven Word?**

**1)vowels=['a','e','i','o','u']**

**3)found=[]**

**5) ifletter in vowels:**

**7)**

**found.append(letter)**

**9)print("Thenumberofdifferent vowelspresentin",word,"is",len(found))**

**8)print(found)**

**ifletternotinfound:**

**6)**

**4)for letter inword:**

**2)word=input("Entertheword tosearchfor vowels: ")**

D:\Python\_classes>pytest.py

Enterthewordtosearchforvowels:durgasoftwaresolutions ['u', 'a', 'o', 'e', 'i']

The number of different vowels present in durgasoftwaresolutions is 5 ListoutallFunctionsofListandwritea ProgramtousetheseFunctions



TUPLE

#### DATASTRUCTURE



**117**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

1. **Tupleisexactlysameas Listexceptthatitisimmutable.i.eoncewecreatesTuple object, we cannot perform any changes in that object.**
2. **HenceTupleisReadonlyversionof List.**
3. **Ifourdataisfixedandneverchangesthenweshouldgo for Tuple.**
4. **InsertionOrderispreserved**
5. **Duplicatesare allowed**
6. **Heterogeneousobjectsareallowed.**
7. **Wecanpreserveinsertionorderandwecandifferentiateduplicateobjectsbyusing index. Hence index will play very important role in Tuple also.**
8. **Tuplesupportboth+veand -veindex.+veindexmeansforwarddirection (fromleftto right) and -ve index means backward direction (from right to left)**
9. **WecanrepresentTupleelementswithinParenthesisandwithcommaseperator.**
10. **Parenethesisareoptionalbutrecommendedtouse.**

1)t=10,20,30,40

|  |
| --- |
| **2)print(t)** |
| **3)print(type(t))** |
| **4)** |
| **5)Output** |
| **6)(10, 20,30,40)**  **7)**  **8)<class'tuple'>** |
| **9) t=()** |
| **10)print(type(t)****tuple** |

Note:Wehavetotakespecialcareaboutsinglevaluedtuple.compulsarythevalue should ends with comma, otherwise it is not treated as tuple.

|  |  |  |
| --- | --- | --- |
|  | **1)** | **t=(10)** |
| **2)** | **print(t)** |
| **3)** | **print(type(t))** |
| **4)** |  |
| **5)** | **Output** |
| **6)** | **10** |
| **7)** | **<class'int'>** |
| **Eg:** |  |  |
|  | **1)** | **t=(10,)** |
|  | **2)** | **print(t)** |
|  | **3)** | **print(type(t))** |
|  | **4)** |  |
|  | **5)** | **Output** |
|  | **6)** | **(10,)** |
|  | **7)** | **<class'tuple'>** |



1. WhichofthefollowingarevalidTuples?
   1. **t=()**
   2. **t =10,20,30,40**
   3. **t=10**
   4. **t=10,**
   5. **t=(10)**
   6. **t=(10,)**
   7. **t = (10,20,30,40)**

TupleCreation:

1. **t = ()**

CreationofEmpty Tuple

1. **t = (10,)**

t=10,

CreationofSinglevaluedTuple,ParenthesisareOptional, shouldendswith Comma

1. **t=10,20, 30**

t = (10,20,30)

CreationofmultivaluesTuples&ParenthesisareOptional.

1. **Byusingtuple() Function:**

1)list=[10,20,30]

|  |
| --- |
| **2) t=tuple(list)** |
| **3) print(t)** |
| **4)** |
| **5) t=tuple(range(10,20,2))** |
| **6) print(t)** |

AccessingElementsofTuple:

Wecan accesseitherbyindexorbysliceoperator

1. **By using Index:**
   1. **t = (10,20,30,40, 50,60)**

**4)print(t[100])****IndexError:tupleindex outofrange**

**2)print(t[0])****10**

3)print(t[-1])60

**118**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



1. **ByusingSlice Operator:**
   1. **t=(10,20,30,40,50,60)**

**4)print(t[::2])**

**2)print(t[2:5])**

3)print(t[2:100])

**Output**

(30,40,50)

(30,40,50,60)

(10,30,50)

TuplevsImmutability:

* **Oncewecreatestuple,wecannot changeits content.**
* **Hencetupleobjectsareimmutable.**

**Eg:**

t =(10,20, 30,40)

t[1]=70TypeError:'tuple'objectdoesnotsupportitem assignment

MathematicalOperatorsforTuple:

Wecan apply+and \*operatorsfor tuple

1. **ConcatenationOperator (+):**
   1. **t1=(10,20,30)**

**4) print(t3)****(10,20,30,40,50,60)**

**2) t2=(40,50,60)**

3) t3=t1+t2

1. **MultiplicationOperatorORRepetitionOperator(\*)**
   1. **t1=(10,20,30)**

**2)t2=t1\*3**

3)print(t2)(10,20,30,10,20,30,10,20,30)



ImportantFunctionsofTuple:

1. **len()**

Toreturnnumberofelements presentinthetuple.

Eg:t=(10,20,30,40)

print(len(t))4

1. **count()**

Toreturnnumberofoccurrencesofgivenelementinthetuple

Eg:t = (10,20,10,10,20)

print(t.count(10))3

1. **index()**
   * **Returnsindexoffirstoccurrenceofthegivenelement.**
   * **Ifthespecifiedelementisnotavailablethenwewillget ValueError.**

Eg:t = (10,20,10,10,20)

print(t.index(10))0

print(t.index(30))ValueError:tuple.index(x):xnotin tuple

1. **sorted()**

Tosortelementsbasedondefaultnaturalsortingorder

1)t=(40,10,30,20)

**4)print(t)**

**2)t1=sorted(t)**

3)print(t1)

**Output**

[10,20,30,40]

(40,10,30,20)

Wecansortaccordingtoreverseofdefaultnaturalsortingorderasfollows t1 = sorted(t, reverse = True)

print(t1)[40,30,20,10]



1. **min()Andmax()Functions:**

Thesefunctionsreturnminandmaxvaluesaccordingtodefaultnaturalsortingorder.

1)t=(40,10,30,20)

**2)print(min(t))****10**

3)print(max(t)) 40

1. **cmp():**

֍Itcomparestheelementsofboth tuples.

֍If bothtuplesareequalthen returns0

֍If thefirsttupleisless than secondtuplethenitreturns-1

֍If thefirsttupleisgreater thansecondtuplethenitreturns+1

1)t1=(10,20,30)

|  |
| --- |
| **2)t2=(40,50,60)** |
| **3)t3=(10,20,30)** |
| **4)print(cmp(t1,t2))****-1** |
| **5)print(cmp(t1,t3))****0** |
| **6)print(cmp(t2,t3))****+1** |

Note:cmp()functionisavailableonlyinPython2 but notinPython3

TuplePackingandUnpacking:

Wecan createatuplebypackingagroup of variables.

**Eg:**

a =10

b=20

c=30

d=40

t =a,b,c,d

print(t)(10,20,30,40)

* **Herea,b, c, darepacked intoaTuplet.ThisisnothingbutTuple packing.**
* **TupleunpackingisthereverseprocessofTuplepacking.**
* **Wecan unpack aTupleandassignitsvaluestodifferent variables.**

1)t=(10,20,30,40)

**2)a,b,c,d=t**

3)print("a=",a,"b=",b,"c=",c,"d=",d)

Output:a= 10 b= 20c=30 d=40



Note:Atthetimeoftupleunpackingthenumberofvariablesandnumberofvalues should be same, otherwise we will get ValueError.

**Eg:**

t=(10,20,30,40)

a,b,c=tValueError:toomanyvaluestounpack(expected 3)

TupleComprehension:

* **TupleComprehensionisnotsupportedbyPython.**
* **t = (x\*\*2forx in range(1,6))**
* **Herewearenot gettingtupleobject andwearegettinggenerator object.**

1)t=(x\*\*2forxinrange(1,6))

**4) print(x)**

**2)print(type(t))**

3)forxin t:

D:\Python\_classes>pytest.py

<class'generator'>1

4

9

16

25

**Q)WriteaProgramtotakeaTuple ofNumbersfromtheKeyboardand Print its Sum and Average?**

1)t=eval(input("Enter Tupleof Numbers:"))

**6)print("The Sum=",sum)**

**4)forxin t:**

**2)l=len(t)**

3)sum=0

5) sum=sum+x

7)print("The Average=",sum/l)

D:\Python\_classes>pytest.py

EnterTupleofNumbers:(10,20,30,40) The Sum= 100

TheAverage=25.0

D:\Python\_classes>pytest.py

EnterTupleofNumbers:(100,200,300)



TheSum=600

TheAverage=200.0

DifferencesbetweenListandTuple:

* **ListandTupleareexactlysame exceptsmalldifference:Listobjectsaremutablewhere as Tuple objects are immutable.**
* **Inbothcasesinsertionorderispreserved, duplicateobjectsareallowed,heterogenous objects are allowed, index and slicing are supported.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **List** |  |  | **Tuple** |  |
| **1)ListisaGroupofCommasepareated Values within Square Brackets and Square Brackets are mandatory.**  **Eg: i= [10,20, 30,40]** | | | **1)TupleisaGroupofCommasepareated Values within Parenthesis and Parenthesis are optional.**  **Eg:t= (10, 20,30,40)**  **t=10,20,30, 40** | | |
| **2)List Objects are Mutable i.e. once we createsListObjectwecanperformany changes in that Object.**  **Eg: i[1]=70** | | | **2)TupleObjecctsareImmutablei.e.once we creates Tuple Object we cannot change its content.**  **t[1]=70****ValueError:tupleobject**  **doesnotsupportitem assignment.** | | |
| **3)IftheContentisnotfixedandkeepon changing then we should go for List.** | | | **3)Ifthecontentisfixedandneverchanges then we should go for Tuple.** | | |
| **4)ListObjectscannotusedasKeysfor Dictionries because Keys should be Hashable and Immutable.** | | | **4)TupleObjectscanbeusedasKeysfor Dictionries because Keys should be Hashable and Immutable.** | | |



#### DATASTRUCTURE

SET



* **Ifwewanttorepresentagroupofunique valuesasasingleentitythenweshouldgo for set.**
* **Duplicatesarenot allowed.**
* **Insertionorderisnotpreserved.Butwecansorttheelements.**
* **Indexingandslicingnotallowedfortheset.**
* **Heterogeneouselementsareallowed.**
* **Setobjectsaremutablei.e oncewecreatessetobjectwecanperformanychangesin that object based on our requirement.**
* **Wecanrepresentsetelementswithincurlybracesandwithcommaseperation**
* **Wecanapplymathematicaloperationslikeunion, intersection,differenceetconset objects.**

CreationofSetObjects:

1)s={10,20,30,40}

**2)print(s)**

**3)print(type(s))**

**Output**

{40,10,20,30}

<class'set'>

Wecan createsetobjectsbyusingset()Functions=set(any sequence)

**Eg1:**

1)l=[10,20,30,40,10,20,10]

**2)s=set(l)**

3)print(s)#{40,10,20, 30}

**Eg2:**

**1)s=set(range(5))**

**2)print(s) #{0,1,2,3,4}**

**Note:**

֍Whilecreatingemptyset wehaveto takespecial care.

֍Compulsoryweshoulduseset()function.

֍s = {}Itistreated asdictionarybutnotemptyset.

1)s={}

**2)print(s)**

3)print(type(s))



**Output**

**{}**

<class'dict'>

**Eg:**

1)s=set()

**2)print(s)**

3)print(type(s))

**Output**

set()

<class'set'>

ImportantFunctionsofSet:

1. **add(x):**

Addsitemxto theset.

* 1. **s={10,20,30}**

**2)s.add(40);**

3)print(s)#{40,10,20, 30}

1. **update(x,y,z):**
   1. **Toaddmultipleitemstotheset.**
   2. **ArgumentsarenotindividualelementsandtheseareIterableobjectslikeList, Range etc.**
   3. **AllelementspresentinthegivenIterableobjectswillbeaddedtotheset.**

1)s={10,20,30}

**4)print(s)**

**2)l=[40,50,60,10]**

3)s.update(l,range(5))

Output:{0,1,2,3,4, 40,10, 50,20,60,30}

**Q)Whatisthedifferencebetweenadd()andupdate()Functions in Set?**

* 1. **Wecanuseadd()toaddindividualitemtotheSet,whereaswecanuseupdate() function to add multiple items to Set.**
  2. **add()functioncantakeonlyoneargumentwhereasupdate()functioncantakeany number of arguments but all arguments should be iterable objects.**



1. **Whichofthefollowingarevalidforsets?**
   1. **s.add(10)**
   2. **s.add(10,20,30)****TypeError:add()takesexactlyoneargument(3given)**
   3. **s.update(10)****TypeError:'int'objectisnot iterable**
   4. **s.update(range(1,10,2),range(0,10,2))**
2. **copy():**
   1. **Returnscopyoftheset.**
   2. **Itiscloned object.**

1)s={10,20,30}

**2)s1=s.copy()**

3)print(s1)

1. **pop():**

Itremovesandreturnssomerandomelementfromtheset.

1)s={40,10,30,20}

**4)print(s)**

**2)print(s)**

3)print(s.pop())

**Output**

{40,10,20,30}

40

{10,20, 30}

1. **remove(x):**
   1. **Itremovesspecifiedelementfromtheset.**
   2. **IfthespecifiedelementnotpresentintheSetthenwewillget KeyError.**

1)s ={40, 10, 30,20}

**4)s.remove(50 KeyError:****)50**

**2)s.remove(30)**

3)print{(s)40,10, 20}

1. **discard(x):**
2. **Itremovesthespecifiedelementfromtheset.**
3. **Ifthespecifiedelementnotpresentintheset thenwewon'tget any error.**

**1)s = {10,20, 30}**

**2)s.discard(10)**



1. **print {****(s)20,30}**

5)print{(s)20,30}

**4)s.discard(50)**

Q)Whatisthedifference betweenremove()anddiscard()functionsinSet?

Q)Explaindifferencesbetweenpop(),remove()anddiscard()functionsinSet?

1. **clear():**

ToremoveallelementsfromtheSet.

1)s={10,20,30}

**4)print(s)**

**2)print(s)**

3)s.clear()

**Output**

{10,20, 30}

set()

###### MathematicalOperationsontheSet:

1. **union():**
   * **x.union(y)****Wecan usethisfunctiontoreturnall elementspresentinbothsets**
   * **x.union(y)ORx|y.**

1)x ={10,20,30,40}

**4)print(x|y)****{10,20,30, 40,50, 60}**

**2)y ={30,40,50,60}**

3)print(x.union(y)){10, 20,30, 40,50,60}

1. **intersection():**
   * **x.intersection(y)ORx&y.**
   * **Returnscommonelementspresentinbothxandy.**

1)x= {10,20, 30,40}

**4)print(x&y)****{40,30}**

**2)y={30,40,50,60}**

3)print (x.intersection(y)){40,30}



1. **difference():**
   * **x.difference(y)ORx-y.**
   * **Returnstheelementspresent inxbutnotin y.**

1)x ={10,20,30,40}

**4)print(x-y)****{10,20}**

**2)y ={30,40,50,60}**

3)print(x.difference(y))10,20

5)print(y-x){50,60}

1. **symmetric\_difference():**
   * **x.symmetric\_difference(y)ORx^y.**
   * **Returnselementspresentin eitherxORybutnotin both.**

1)x ={10,20,30,40}

2)y ={30,40,50,60}

3)print (x.symmetric\_difference(y)){10,50,20, 60}

4)print(x^y){10, 50,20,60}

MembershipOperators:(in,notin)

1)s=set("durga")

**4)print('z'ins)**

**2)print(s)**

3)print('d'ins)

**Output**

{'u','g','r','d','a'}

True False

SetComprehension:

Setcomprehensionispossible.

1)s={x\*xforxinrange(5)}

**4)s={2\*\*xforxinrange(2,10,2)}**

**2)print(s)****{0,1,4,9,16}**

3)

5)print(s){16,256,64,4}



SetObjectswon'tsupportindexingandslicing:

1)s={10,20,30,40}

**2)print(s[0])****TypeError:'set'objectdoesnotsupport indexing**

3)print(s[1:3])TypeError:'set'objectisnotsubscriptable

**Q)WriteaProgramtoeliminateDuplicatesPresentintheList?**

|  |  |  |  |
| --- | --- | --- | --- |
| **Approach-1**   1. **l=eval(input("EnterListofvalues:"))** 2. **s=set(l)** 3. **print(s)**   **D:\Python\_classes>pytest.py**  **EnterListofvalues: [10,20,30,10,20,40]**  **{40,10,20,30}** |  |  | **Approach-2**   1. **l=eval(input("EnterListofvalues:"))** 2. **l1=[]** 3. **forxin l:** 4. **ifxnot in l1:** 5. **l1.append(x)** 6. **print(l1)**   **D:\Python\_classes>py test.py**  **EnterListofvalues:[10,20,30,10,20,40] [10, 20, 30, 40]** |

**Q)WriteaProgramtoPrintdifferentVowelsPresentinthegiven Word?**

1)w=input("Enterwordtosearchfor vowels:")

**4)d=s.intersection(v)**

**2)s=set(w)**

3)v={'a','e','i','o','u'}

5)print("Thedifferent vowel presentin",w,"are",d)

D:\Python\_classes>pytest.py

Enterwordtosearchforvowels:durga

Thedifferentvowelpresentindurgaare{'u','a'}



### DICTIONARY

#### DATASTRUCTURE



֍WecanuseList,TupleandSettorepresentagroup ofindividualobjectsasasingle entity.

֍Ifwewanttorepresent agroupofobjectsaskey-valuepairsthenweshouldgofor Dictionary.

**Eg:**

* **rollno name**
* **phonenumber--address**
* **ipaddress domainname**

֍Duplicatekeysarenotallowedbut valuescanbeduplicated.

֍Hetrogeneousobjectsareallowedfor bothkeyand values.

֍Insertionorderisnot preserved

֍Dictionariesaremutable

֍Dictionariesaredynamic

֍indexingandslicingconceptsarenotapplicable

Note:InC++andJavaDictionariesareknownas"Map"whereasinPerlandRubyitis known as "Hash"

HowtoCreateDictionary?

* **d={} OR d=dict()**
* **Wearecreatingemptydictionary.Wecanaddentriesasfollows**

1)d[100]="durga"

**4)print(d)****{100:'durga',200: 'ravi',300:'shiva'}**

**2)d[200]="ravi"**

3)d[300]="shiva"

* **Ifweknow datainadvancethenwecancreatedictionaryasfollows**
* **d={100:'durga',200:'ravi', 300:'shiva'}**
* **d={key:value,key:value}**

Howto AccessDatafromtheDictionary?

Wecan access databyusingkeys.

1)d={100:'durga',200:'ravi', 300:'shiva'}

**2)print(d[100]) #durga**

3)print(d[300]) #shiva

Ifthespecifiedkeyisnot availablethenwewillget KeyError



print(d[400])KeyError:400

Wecanpreventthisbycheckingwhetherkeyisalreadyavailableornotbyusing has\_key() function or by using in operator.

d.has\_key(400)Returns1ifkeyisavailableotherwisereturns0

Buthas\_key()functionisavailableonlyin Python2butnotinPython3.Hence compulsory we have to use in operator.

if400 in d:

print(d[400])

1. **WriteaProgramtoEnterNameandPercentageMarksin a Dictionary and Display Information on the Screen**
   1. **rec={}**

**10)for xinrec:**

**8) i=i+1**

**6) marks=input("Enter%of MarksofStudent:")**

**4)while i<=n:**

**2)n=int(input("Enternumber ofstudents: "))**

3)i=1

5) name=input("EnterStudentName:")

7) rec[name]=marks

9)print("Nameof Student","\t","%of marks")

11) print("\t",x,"\t\t",rec[x])

D:\Python\_classes>pytest.py Enter number of students: 3 Enter Student Name: durga

Enter%ofMarksofStudent:60% Enter Student Name: ravi

Enter%ofMarksofStudent:70% Enter Student Name: shiva

Enter %ofMarks of Student:80%

Name of Student % of marks

**--------------- ------------**

durga 60%

ravi 70 %

shiva 80%



HowtoUpdateDictionaries?

֍d[key]= value

֍Ifthekeyisnotavailablethenanewentrywillbeaddedtothedictionarywiththe specified key-value pair

֍Ifthekeyisalreadyavailablethenoldvaluewillbereplacedwithnew value.

1)d={100:"durga",200:"ravi",300:"shiva"}

|  |
| --- |
| **2)print(d)** |
| **3)d[400]="pavan"** |
| **4)print(d)** |
| **5)d[100]="sunny"** |
| **6)print(d)** |

**Output**

{100:'durga',200:'ravi',300:'shiva'}

{100:'durga',200:'ravi',300:'shiva',400:'pavan'}

{100:'sunny',200:'ravi',300:'shiva',400:'pavan'}

HowtoDeleteElementsfromDictionary?

1. **deld[key]**
   * **Itdeletesentryassociatedwiththespecifiedkey.**
   * **Ifthekeyisnotavailablethenwewillget KeyError.**

1)d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d)**

**2)print(d)**

3)deld[100]

5)deld[400]

**Output**

{100:'durga',200:'ravi',300:'shiva'}

{200:'ravi',300:'shiva'}

KeyError:400

1. **d.clear()**

Toremoveallentriesfromthedictionary.

1) d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d)**

**2)print(d)**

3)d.clear()



**Output**

{100:'durga',200:'ravi',300:'shiva'}

**{}**

1. **deld**

Todeletetotaldictionary.Nowwecannotaccessd.

1)d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d)**

**2)print(d)**

3)deld

**Output**

{100:'durga',200:'ravi',300:'shiva'} NameError: name 'd'is not defined

ImportantFunctionsofDictionary:

1. **dict():**

Tocreatea dictionary

* + **d=dict()****Itcreatesemptydictionary**
  + **d=dict({100:"durga",200:"ravi"})****Itcreatesdictionarywith specifiedelements**
  + **d= dict([(100,"durga"),(200,"shiva"),(300,"ravi")])**

Itcreatesdictionarywiththegivenlistoftuple elements

1. **len()**

Returnsthenumberof itemsinthe dictionary.

1. **clear():**

Toremoveallelementsfromthedictionary.

1. **get():**

Toget thevalueassociatedwith thekey

**d.get(key)**

IfthekeyisavailablethenreturnsthecorrespondingvalueotherwisereturnsNone.It wont raise any error.



**d.get(key,defaultvalue)**

Ifthekeyisavailablethenreturnsthecorrespondingvalueotherwisereturnsdefault value.

1)d={100:"durga",200:"ravi",300:"shiva"}

**6)print(d.get(100,"Guest"))****durga**

**4)print(d.get(100))****durga**

**2)print(d[100])** **durga**

3)print(d[400]) KeyError:400

5)print(d.get(400))None

7)print(d.get(400,"Guest"))Guest

1. **pop():**

d.pop(key)

* + **Itremovestheentryassociatedwiththespecifiedkeyandreturnsthe corresponding value.**
  + **Ifthespecifiedkeyisnotavailablethenwewillget KeyError.**

1)d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d.pop(400))**

**2)print(d.pop(100))**

3)print(d)

**Output**

durga

{200:'ravi',300:'shiva'}

KeyError:400

1. **popitem():**

Itremovesanarbitraryitem(key-value)fromthedictionatyandreturnsit.

1)d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d)**

**2)print(d)**

3)print(d.popitem())

**Output**

{100:'durga',200:'ravi',300:'shiva'}

(300,'shiva')

{100:'durga',200:'ravi'}

IfthedictionaryisemptythenwewillgetKeyError d={}

print(d.popitem())==>KeyError:'popitem():dictionaryisempty'



1. **keys():**

Itreturnsallkeysassociatedeith dictionary.

1)d={100:"durga",200:"ravi",300:"shiva"}

**4) print(k)**

**2)print(d.keys())**

3)forkin d.keys():

**Output**

dict\_keys([100,200, 300])

100

200

300

1. **values():**

Itreturnsallvaluesassociatedwiththe dictionary.

1)d={100:"durga",200:"ravi",300:"shiva"}

**4) print(v)**

**2)print(d.values())**

3)forvind.values():

**Output**

dict\_values(['durga','ravi','shiva']) durga

ravi shiva

1. **items():**

Itreturnslistoftuplesrepresentingkey-valuepairs. [(k,v),(k,v),(k,v)]

1)d={100:"durga",200:"ravi",300:"shiva"}

**2)fork,vind.items():**

3) print(k,"--",v)

**Output**

100--durga

200--ravi

300--shiva



1. **copy():**

Tocreateexactlyduplicatedictionary(clonedcopy) d1 = d.copy();

1. **setdefault():**

d.setdefault(k,v)

* + **Ifthekeyisalreadyavailablethenthisfunctionreturnsthecorrespondingvalue.**
  + **Ifthekeyisnotavailablethenthespecifiedkey-valuewillbeaddedasnewitemto the dictionary.**

1)d={100:"durga",200:"ravi",300:"shiva"}

**4)print(d.setdefault(100,"sachin"))**

**2)print(d.setdefault(400,"pavan"))**

3)print(d)

5)print(d)

**Output**

pavan

{100:'durga',200:'ravi',300:'shiva',400:'pavan'} durga

{100:'durga',200:'ravi',300:'shiva',400:'pavan'}

1. **update():**

d.update(x)

Allitemspresentinthedictionaryxwillbeaddedtodictionaryd

1. **WriteaProgramtotakeDictionaryfromtheKeyboardand print the Sum of Values?**
   1. **d=eval(input("Enterdictionary:"))**

**2)s=sum(d.values())**

3)print("Sum= ",s)

**Output**

D:\Python\_classes>pytest.py

Enterdictionary:{'A':100,'B':200,'C':300} Sum=600



1. **WriteaProgramtofindNumberofOccurrencesofeachLetterpresent in the given String?**
   1. **word=input("Enteranyword:")**

|  |
| --- |
| **2)d={}** |
| **3)forxinword:** |
| **4) d[x]=d.get(x,0)+1** |
| **5)fork,v ind.items():** |
| **6) print(k,"occurred",v,"times")** |

Output D:\Python\_classes>pytest.py Enter any word: mississippim occurred1times

i occurred 4 timess occurred 4 timespoccurred2times

**Q)WriteaProgramtofindNumberofOccurrencesofeachVowelpresent in the given String?**

1)word=input("Enteranyword: ")

|  |  |
| --- | --- |
| **2)** | **vowels={'a','e','i','o','u'}** |
| **3)** | **d={}** |
| **4)forxinword:** | |
| **5) ifxinvowels:** | |
| **6)** | **d[x]=d.get(x,0)+1** |
| **7)fork,vinsorted(d.items()):** | |
| **8) print(k,"occurred",v,"times")** | |

**Output**

D:\Python\_classes>pytest.py

Enteranyword:doganimaldoganimal a occurred4times

i occurred2times ooccurred2times



1. **Write a Program to accept Student Name and Marks from theKeyboardandcreatesaDictionary.AlsodisplayStudentMarksby taking Student Name as Input?**

**1)n=int(input("Enterthenumberofstudents:"))**

**3)for iinrange(n):**

**5) marks=input("EnterStudentMarks:")**

**7)whileTrue:**

**9) marks=d.get(name,-1)**

**11)**

**print("StudentNotFound")**

**13)**

**print("TheMarksof",name,"are",marks)**

**15) ifoption=="No":**

**17)print("Thanksforusingourapplication")**

**break**

**16)**

**14) option=input("Doyouwanttofind anotherstudent marks[Yes|No]")**

**12) else:**

**10) ifmarks==-1:**

**8) name=input("EnterStudentNametogetMarks:")**

**6) d[name]=marks**

**4) name=input("EnterStudentName:")**

**2)d={}**

Output D:\Python\_classes>py test.py Enterthenumberofstudents:5

EnterStudentName:sunny Enter Student Marks: 90

EnterStudentName:banny Enter Student Marks: 80

EnterStudentName:chinny Enter Student Marks: 70

EnterStudentName:pinny Enter Student Marks: 60

EnterStudentName:vinny Enter Student Marks: 50

EnterStudentNametogetMarks:sunny The Marks of sunny are 90



Doyouwanttofindanotherstudentmarks[Yes|No]Yes Enter Student Name to get Marks: durga

StudentNotFound

Doyouwanttofindanotherstudentmarks[Yes|No]No Thanks for using our application

Dictionary Comprehension:

Comprehensionconceptapplicablefordictionariesalso.

1)squares={x:x\*xforxinrange(1,6)}

**4)print(doubles)**

**2)print(squares)**

3)doubles={x:2\*xforxinrange(1,6)}

**Output**

{1: 1,2:4,3:9,4:16, 5:25}

{1: 2,2:4,3:6,4:8,5:10}



**142**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

## FUNCTIONS

֍If a group of statements is repeatedly required then it is not recommended to write thesestatementseverytimeseperately.Wehavetodefinethesestatementsasasingle unit and we can call that unit any number of times based on our requirement without rewriting. This unit is nothing but function.

֍Themainadvantageoffunctionsiscode Reusability.

֍Note:Inotherlanguagesfunctionsareknownasmethods,procedures,subroutinesetc

֍Pythonsupports2typesoffunctions

1. **BuiltinFunctions**
2. **UserDefinedFunctions**
   1. Builtin Functions:

Thefunctionswhicharecomingalongwith Pythonsoftwareautomatically, arecalled built in functions or pre defined functions.

**Eg:id()**

type() input() eval()

etc..

* 1. UserDefined Functions:

Thefunctionswhicharedevelopedbyprogrammerexplicitlyaccordingtobusiness requirements, are called user defined functions.

**SyntaxtoCreateUserdefinedFunctions:**

def function\_name(parameters): """ doc string"""

**----**

**-----**

returnvalue

Note:Whilecreatingfunctionswecanuse2 keywords

* + 1. **def (mandatory)**
    2. **return(optional)**

Eg1:WriteafunctiontoprintHello

**test.py**

1)defwish():

**2) print("HelloGoodMorning")**

3)wish()



5)wish()

**4)wish()**

##### Parameters

Parametersareinputstothefunction.Ifafunctioncontainsparameters, thenatthetime of calling,compulsory we should provide values otherwise,otherwise we will geterror.

Eg:Writeafunctiontotakenameofthestudentasinputandprintwishmessageby name.

**1)defwish(name):**

**3)wish("Durga")**

**4)wish("Ravi")**

**print("Hello",name,"GoodMorning")**

**2)**

D:\Python\_classes>pytest.py Hello DurgaGood Morning Hello RaviGood Morning

Eg:Writeafunctiontotakenumberasinputandprintitssquarevalue

1)defsquareIt(number):

**4)squareIt(5)**

**2) print("TheSquareof",number,"is",number\*number)**

3)squareIt(4)

D:\Python\_classes>pytest.py The Square of 4 is 16

TheSquareof5is25

##### ReturnStatement:

Functioncantakeinputvaluesasparametersandexecutesbusinesslogic, andreturns output to the caller with return statement.

**Q)WriteaFunctiontoaccept2NumbersasInputandreturn Sum**

1)defadd(x,y):

**4)print("Thesumis",result)**

**2) returnx+y**

3)result=add(10,20)

5)print("Thesumis",add(100,200))



D:\Python\_classes>pytest.py The sum is 30

The sumis300

IfwearenotwritingreturnstatementthendefaultreturnvalueisNone.

1)deff1():

**4)print(f1())**

**2) print("Hello")**

3)f1()

**OutputHello Hello None**

**Q)WriteaFunctiontocheckwhetherthegivenNumberisEven OR Odd?**

1)defeven\_odd(num):

**6)even\_odd(10)**

**4) else:**

**2) ifnum%2==0:**

3) print(num,"isEvenNumber")

5) print(num,"isOddNumber")

7)even\_odd(15)

Output D:\Python\_classes>pytest.py 10 is Even Number

15is Odd Number

1. **WriteaFunctiontofindFactorialofgivenNumber?**

1)deffact(num):

|  |
| --- |
| **2) result=1** |
| **3) whilenum>=1:** |
| **4) result=result\*num** |
| **5) num=num-1** |
| **6) returnresult** |
| **7)for iinrange(1,5):** |
| **8) print("TheFactorialof",i,"is:",fact(i))** |



Output D:\Python\_classes>pytest.py The Factorial of 1 is : 1

The Factorial of 2 is : 2 The Factorial of 3 is : 6 TheFactorialof4is:24

ReturningMultipleValuesfromaFunction:

InotherlanguageslikeC,C++andJava,functioncanreturnatmostonevalue.Butin Python, a function can return any number of values.

**Eg1:**

1)defsum\_sub(a,b):

**6)print("TheSum is :",x)**

**4) returnsum,sub**

**2) sum=a+b**

3) sub=a-b

5)x,y=sum\_sub(100,50)

7)print("TheSubtractionis:",y)

**Output**

TheSumis:150

TheSubtractionis:50

**Eg2:**

1. **defcalc(a,b):**

|  |
| --- |
| **2) sum=a+b** |
| **3) sub=a-b** |
| **4) mul=a\*b** |
| **5) div=a/b** |
| **6) returnsum,sub,mul,div** |
| **7)t=calc(100,50)** |
| **8)print("TheResultsare")** |
| **9)for iint:** |
| **10) print(i)** |

**Output**

TheResultsare 150

50

5000

2.0



TypesofArguments

deff1(a,b):

**------**

**------**

------ f1(10,20)

a,bareformalargumentswhereas10,20areactualarguments. There are 4 types are actual arguments are allowed in Python.

* 1. **Positional Arguments**
  2. **KeywordArguments**
  3. **DefaultArguments**
  4. **VariableLengthArguments**
  5. PositionalArguments:
     + **Thesearetheargumentspassedtofunctionincorrectpositionalorder. def sub(a, b):**

print(a-b)

sub(100, 200)

sub(200, 100)

* + - **Thenumberofargumentsandpositionofarguments mustbematched.Ifwechange the order then result may be changed.**
    - **Ifwechangethenumber ofargumentsthenwewill geterror.**
  1. KeywordArguments:

Wecanpassargumentvaluesbykeywordi.ebyparametername.

1)defwish(name,msg):

**4)wish(msg="GoodMorning",name="Durga")**

**2) print("Hello",name,msg)**

3)wish(name="Durga",msg="GoodMorning")

**Output**

HelloDurgaGoodMorning Hello DurgaGood Morning

Heretheorderofargumentsisnotimportantbutnumberofargumentsmustbematched.



Note:Wecanusebothpositionalandkeywordargumentssimultaneously.Butfirstwe have to take positional arguments and then keyword arguments,otherwise we will get syntaxerror.

1)defwish(name,msg):

**4)wish("Durga",msg="GoodMorning")****Valid**

**2) print("Hello",name,msg)**

3)wish("Durga","GoodMorning")Valid

1. **wish(name="Durga","GoodMorning")****Invalid**
2. **SyntaxError:positionalargumentfollowskeywordargument**
   1. DefaultArguments:

Sometimeswecanprovidedefaultvaluesforourpositionalarguments.

1. **defwish(name="Guest"):**

**3)wish("Durga")**

1. **print("Hello",name,"GoodMorning")**

**4)wish()**

**Output**

HelloDurgaGoodMorning HelloGuestGoodMorning

Ifwearenotpassinganynamethenonlydefaultvaluewillbeconsidered.

**\*\*\*Note:**

Afterdefaultargumentsweshouldnottakenondefaultarguments.

1)defwish(name="Guest",msg="GoodMorning"):Valid

**2)defwish(name,msg="GoodMorning"):****Valid**

3)defwish(name="Guest",msg):Invalid

SyntaxError:non-defaultargumentfollowsdefaultargument

* 1. VariableLengthArguments:
     + **Sometimeswecanpassvariablenumberofargumentstoourfunction, suchtypeof arguments are called variable length arguments.**
     + **Wecandeclareavariablelength argumentwith\* symbolasfollows**
     + **def f1(\*n):**
     + **Wecancallthisfunctionbypassinganynumberofargumentsincludingzeronumber.**
     + **Internallyallthesevaluesrepresentedintheformoftuple.**



1)defsum(\*n):

|  |
| --- |
| **2) total=0** |
| **3) forn1inn:** |
| **4) total=total+n1** |
| **5) print("TheSum=",total)** |
| **6)** |
| **7)sum()** |
| **8)sum(10)** |
| **9)sum(10,20)** |
| **10)sum(10,20,30,40)** |

**Output**

TheSum=0

TheSum=10

TheSum=30

TheSum=100

Note:Wecanmixvariablelengthargumentswithpositionalarguments.

1)deff1(n1,\*s):

|  |
| --- |
| **2) print(n1)** |
| **3) for s1in s:** |
| **4) print(s1)** |
| **5)** |
| **6)f1(10)** |
| **7)f1(10,20,30,40)** |
| **8)f1(10,"A",30,"B")** |

**Output**

10

10

20

30

40

10

A 30 B

Note:Aftervariablelengthargument,ifwearetakinganyotherargumentsthenwe should provide values as keyword arguments.



1)deff1(\*s,n1):

|  |
| --- |
| **2) for s1in s:** |
| **3) print(s1)** |
| **4) print(n1)** |
| **5)** |
| **6)f1("A","B",n1=10)** |

**Output**

A B 10

f1("A","B",10)Invalid

TypeError:f1()missing1requiredkeyword-onlyargument: 'n1'

Note:Wecandeclarekeywordvariablelengtharguments also.

* + - **Forthiswehavetouse\*\*.**
    - **deff1(\*\*n):**
    - **Wecancallthisfunctionbypassinganynumberofkeywordarguments. Internally these keyword arguments will be stored inside a dictionary.**

1)defdisplay(\*\*kwargs):

**4)display(n1=10,n2=20,n3=30)**

**2) for k,vin kwargs.items():**

3) print(k,"=",v)

5)display(rno=100,name="Durga",marks=70,subject="Java")

**Output**

n1=10

n2=20

n3=30

rno = 100 name=Durga marks = 70 subject=Java



###### Case Study:

def f(arg1,arg2,arg3=4,arg4=8): print(arg1,arg2,arg3,arg4)

1. **f(3,2)****3248**
2. **f(10,20,30,40)****10203040**
3. **f(25,50,arg4=100)****25504100**
4. **f(arg4=2,arg1=3,arg2=4)****3442**
5. **f()** **Invalid**

TypeError:f()missing2requiredpositionalarguments:'arg1'and'arg2'

1. **f(arg3=10,arg4=20, 30,40)****Invalid**

SyntaxError:positionalargumentfollowskeywordargument

[Afterkeywordargumentsweshouldnottakepositional arguments]

1. **f(4, 5, arg2= 6)****Invalid**

TypeError:f()gotmultiplevaluesforargument'arg2'

1. **f(4, 5, arg3= 5, arg5=6)****Invalid**

TypeError:f()gotanunexpectedkeywordargument'arg5'

Note:Function vsModulevsLibrary

1. **Agroup oflineswithsomenameiscalled afunction**
2. **Agroupoffunctionssavedtoafile,iscalled Module**
3. **AgroupofModulesisnothingbutLibrary**

Library

Function

|  |
| --- |
| **Module 1 Module2**  **Function1 Function1**  **Function2 Function2**  **Function3 Function3** |

**-----------------**

**-----------------**

**-----------------**

**-----------------**

**-----------------**

**-----------------**

**-----------------**

**-----------------**

**-----------------**



TypesofVariables

Python supports2typesof variables.

1. **Global Variables**
2. **Local Variables**
3. **Global Variables**
   * **Thevariableswhicharedeclaredoutsideoffunctionarecalledglobalvariables.**
   * **Thesevariablescanbeaccessedinallfunctionsofthatmodule.**

**1)a=10#globalvariable**

**3) print(a)**

**5)deff2():**

**7)**

**9)f2()**

**8)f1()**

**6) print(a)**

**4)**

**2)deff1():**

**Output**

10

10

1. **LocalVariables:**
   * **Thevariableswhicharedeclaredinsideafunctionarecalledlocalvariables.**
   * **Localvariablesareavailableonlyforthe functioninwhichwedeclaredit.i.efrom outside of function we cannot access.**

**1)deff1():**

**3) print(a) #valid**

**5)deff2():**

**7)**

**9)f2()**

**8)f1()**

**6) print(a) #invalid**

**4)**

**2) a=10**

NameError:name'a'isnot defined



globalKeyword:

Wecanuseglobalkeywordforthefollowing2 purposes:

1. **Todeclareglobalvariableinsidefunction**
2. **Tomakeglobalvariableavailabletothefunctionsothatwecanperformrequired modifications**

**1)a=10**

**3) a=777**

**5)**

**7) print(a)**

**9)f1()**

**11)**

**10)f2()**

**8)**

**6)deff2():**

**4) print(a)**

**2)deff1():**

**Output**

777

10

1)a=10

**9)f1()**

**7) print(a)**

**4) a=777**

**2)deff1():**

3) globala

1. **print(a)**
2. **deff2():**

8)

10)f2()

**Output**

777

777

**1)deff1():**

**3) print(a)**

**5)deff2():**

**7)**

**6) print(a)**

**4)**

**2) a=10**



9)f2()

**8)f1()**

Output:NameError:name'a'isnotdefined

1)deff1():

|  |
| --- |
| **2) globala** |
| **3) a=10** |
| **4) print(a)** |
| **5)** |
| **6)deff2():** |
| **7) print(a)** |
| **8)** |
| **9)f1()** |
| **10)f2()** |

**Output**

10

10

Note:Ifglobalvariableandlocalvariablehavingthesamenamethenwecanaccess global variable inside a function as follows

1)a=10GlobalVariable

|  |
| --- |
| **2)deff1():** |
| **3) a=777****LocalVariable** |
| **4) print(a)** |
| **5) print(globals()['a'])** |
| **6)f1()** |

**Output**

777

10

RecursiveFunctions

AfunctionthatcallsitselfisknownasRecursiveFunction.

**Eg:**

factorial(3)= 3\*factorial(2)

=3\*2\*factorial(1)

=3\*2\*1\*factorial(0)

=3\*2\*1\*1

=6

factorial(n)=n\*factorial(n-1)



Themainadvantagesofrecursivefunctions are:

1. **Wecanreducelengthofthecodeandimproves readability.**
2. **Wecansolvecomplexproblemsvery easily.**
3. **WriteaPythonFunctiontofindFactorialofgivenNumber with Recursion**
   1. **deffactorial(n):**

|  |
| --- |
| **2) ifn==0:** |
| **3) result=1** |
| **4) else:** |
| **5) result=n\*factorial(n-1)** |
| **6) returnresult** |
| **7)print("Factorial of4is:",factorial(4))** |
| **8)print("Factorial of5is:",factorial(5))** |

**Output**

Factorialof4is:24

Factorialof5is:120

AnonymousFunctions:

* **Sometimeswecandeclareafunctionwithoutanyname,suchtypeofnameless functions are called anonymous functions or lambda functions.**
* **Themainpurposeofanonymousfunctionisjustforinstantuse(i.eforonetime usage)**

NormalFunction:

Wecandefinebyusingdef keyword.

**defsquareIt(n):return n\*n**

LambdaFunction:

**Wecandefinebyusinglambda keyword lambdan:n\*n**

**SyntaxoflambdaFunction:lambdaargument\_list:expression**

Note:ByusingLambdaFunctionswecanwriteveryconcise codesothatreadabilityof the program will be improved.



**Q)WriteaProgramtocreateaLambdaFunctiontofindSquareofgiven Number?**

1)s=lambdan:n\*n

**2)print("TheSquareof4 is:",s(4))**

3)print("TheSquareof5 is:",s(5))

**Output**

TheSquareof4is:16 TheSquareof5is: 25

**Q)LambdaFunctiontofindSumof2givenNumbers**

1)s=lambdaa,b:a+b

**2)print("TheSum of10,20is:",s(10,20))**

3)print("TheSum of100,200 is:",s(100,200))

**Output**

TheSumof 10,20is: 30

TheSumof100,200is:300

1. **LambdaFunctiontofindbiggestofgiven Values**
   1. **s=lambdaa,b:aifa>belseb**

**2)print("TheBiggestof10,20 is:",s(10,20))**

3)print("TheBiggestof100,200 is:",s(100,200))

**Output**

TheBiggest of10,20is:20

TheBiggest of100,200is: 200

Note:LambdaFunctioninternallyreturnsexpressionvalueandwearenotrequiredto write return statement explicitly.

Note:Sometimeswecanpassfunctionasargumenttoanotherfunction.Insuchcases lambda functions are best choice.

Wecanuselambdafunctionsverycommonlywithfilter(), map()andreduce()functions, because these functions expect function as argument.



filter() Function:

Wecanusefilter()functiontofiltervaluesfromthegivensequencebasedonsome condition.

filter(function,sequence)

WhereFunctionArgumentisresponsibletoperformconditionalcheck Sequencecanbe List OR Tuple OR String.

**Q)ProgramtofilteronlyEvenNumbersfromtheListbyusingfilter() Function?**

**WithoutLambdaFunction:**

1)defisEven(x):

|  |
| --- |
| **2) ifx%2==0:** |
| **3) returnTrue** |
| **4) else:** |
| **5) returnFalse** |
| **6)l=[0,5,10,15,20,25,30]** |
| **7)l1=list(filter(isEven,l))** |
| **8)print(l1)#[0,10,20,30]** |

**WithLambdaFunction:**

1)l=[0,5,10,15,20,25,30]

**4)l2=list(filter(lambdax:x%2!=0,l))**

**2)l1=list(filter(lambdax:x%2==0,l))**

3)print(l1)#[0,10,20,30]

5)print(l2)#[5,15,25]

map() Function:

* **For every element present in the given sequence,apply some functionality and generatenewelementwiththerequiredmodification.Forthisrequirementwe should go for map() function.**
* **Eg:Foreveryelementpresentinthelistperformdoubleandgeneratenewlistof doubles.**
* **Syntax:map(function,sequence)**
* **Thefunctioncanbeappliedoneachelementofsequenceandgeneratesnew sequence.**



**WithoutLambda**

1) l=[1,2,3,4,5]

**4)l1=list(map(doubleIt,l))**

**2)defdoubleIt(x):**

3) return2\*x

5)print(l1)#[2,4,6,8,10]

**WithLambda**

1)l=[1,2,3,4,5]

**2)l1=list(map(lambdax:2\*x,l))**

3)print(l1) #[2,4,6,8,10]

**-------------------------------------------------------------**

Eg2:Tofindsquareofgiven numbers

1)l=[1,2,3,4,5]

**2)l1=list(map(lambdax:x\*x,l))**

3)print(l1) #[1,4,9,16,25]

Wecanapplymap()functiononmultiplelistsalso.Butmakesurealllistshouldhavesame length.

Syntax:map(lambdax,y:x\*y,l1,l2)) xisfroml1and yisfrom l2

1)l1=[1,2,3,4]

**4)print(l3) #[2,6,12,20]**

**2)l2=[2,3,4,5]**

3)l3=list(map(lambdax,y:x\*y,l1,l2))

reduce()Function:

* **reduce()functionreducessequenceofelementsintoasingleelementbyapplyingthe specified function.**
* **reduce(function,sequence)**
* **reduce()functionpresentinfunctoolsmoduleandhenceweshouldwrite import statement.**

1)fromfunctoolsimport\*

**4)print(result)#150**

**2)l=[10,20,30,40,50]**

3)result=reduce(lambdax,y:x+y,l)



**Eg:**

**1)result=reduce(lambdax,y:x\*y,l)**

**2)print(result)#12000000**

**Eg:**

1)fromfunctoolsimport\*

**2)result=reduce(lambdax,y:x+y,range(1,101))**

3)print(result)#5050

EverythingisanObject:

* **InPythoneverythingistreatedasobject.**
* **Evenfunctionsalsointernallytreatedasobjectsonly.**

1)deff1():

**4)print(id(f1))**

**2) print("Hello")**

3)print(f1)

**Output:**

<functionf1at0x00419618>4298264

FunctionAliasing:

Fortheexistingfunctionwecangiveanothername,whichisnothingbutfunctionaliasing.

1)defwish(name):

**8)greeting('Durga')**

**6)print(id(greeting))**

**4)greeting=wish**

**2) print("GoodMorning:",name)**

3)

5)print(id(wish))

7)

9)wish('Durga')

**Output:**

4429336

4429336

GoodMorning:Durga GoodMorning:Durga



**Note:**

* **Intheaboveexampleonlyonefunctionisavailablebutwecancallthatfunctionby using either wish name or greeting name.**
* **Ifwedeleteonenamestillwecan accessthatfunctionbyusingaliasname.**

**1)defwish(name):**

**3)**

**5)**

**7)wish('Durga')**

**9)delwish**

**11)greeting('Pavan')**

**10)#wish('Durga')****NameError:name'wish'isnot defined**

**8)**

**6)greeting('Durga')**

**4)greeting=wish**

**2) print("GoodMorning:",name)**

**Output:**

GoodMorning:Durga GoodMorning:Durga GoodMorning:Pavan

NestedFunctions:

Wecandeclareafunctioninsideanotherfunction,suchtypeoffunctionsarecalledNested functions.

1)defouter():

|  |
| --- |
| **2) print("outerfunctionstarted")** |
| **3) definner():** |
| **4) print("innerfunctionexecution")** |
| **5) print("outerfunctioncallinginnerfunction")** |
| **6) inner()** |
| **7)outer()** |
| **8)#inner()****NameError:name'inner' isnotdefined** |

**Output:**

outerfunctionstarted

outerfunctioncallinginnerfunction inner function execution

Intheaboveexampleinner()functionislocaltoouter()functionandhenceitisnot possible to call directly from outside of outer() function.

Note:Afunctioncanreturnanotherfunction.



1) defouter():

|  |
| --- |
| **2) print("outerfunctionstarted")** |
| **3) definner():** |
| **4) print("innerfunctionexecution")** |
| **5) print("outerfunctionreturninginnerfunction")** |
| **6) returninner** |
| **7)f1=outer()** |
| **8)f1()** |
| **9)f1()** |
| **10)f1()** |

**Output:**

outerfunctionstarted

outerfunctionreturninginnerfunction inner function execution

innerfunctionexecution innerfunctionexecution

1. **Whatisthedifferenecebetweenthefollowinglines?**

f1 = outer f1=outer()

* **Inthefirstcasefortheouter()functionweareprovidinganothernamef1 (function aliasing).**
* **Butinthesecondcasewecallingouter()function,whichreturnsinnerfunction.For that inner function() we are providing another name f1**

Note:Wecanpassfunctionasargumentto another function

Eg:filter(function,sequence) map(function,sequence) reduce(function,sequence)



MODULES



* **Agroupoffunctions, variablesandclassessavedtoafile,whichisnothingbut module.**
* **EveryPython file(.py)actsasamodule.**

**durgamath.py**

**1)x=888**

**3)defadd(a,b):**

**5)**

**7) print("TheProduct:",a\*b)**

**6)defproduct(a,b):**

**4) print("TheSum:",a+b)**

**2)**

* **durgamathmodulecontainsonevariableand2 functions.**
* **Ifwewanttousemembersofmoduleinourprogramthenweshould importthat module.**

import modulename

* **Wecanaccessmembersbyusingmodulename. modulename.variable**

modulename.function()

**test.py:**

1)importdurgamath

**4)durgamath.product(10,20)**

**2)print(durgamath.x)**

3)durgamath.add(10,20)

**Output**

888

The Sum:30

TheProduct:200

Note:Wheneverweareusing amoduleinourprogram,forthatmodulecompiledfile will be generated and stored in the hard disk permanently.



RenamingaModuleatthetimeofimport(Module Aliasing):

* **Eg:importdurgamathasm**
* **Heredurgamath isoriginalmodulenameandmisalias name.**
* **Wecanaccessmembersbyusingaliasnamem**

**test.py:**

1)importdurgamathasm

**4)m.product(10,20)**

**2)print(m.x)**

3)m.add(10,20)

from... import:

Wecanimportparticularmembersofmodulebyusingfrom...import.

Themainadvantageofthisiswecanaccessmembersdirectlywithoutusingmodule name.

1)from durgamathimportx,add

**4)product(10,20)****NameError:name'product'isnotdefined**

**2)print(x)**

3)add(10,20)

Wecanimportallmembersofamoduleasfollowsfromdurgamath import\*

**test.py:**

1)from durgamathimport\*

**4)product(10,20)**

**2)print(x)**

3)add(10,20)

VariousPossibiltiesofimport:

* 1. **import modulename**
  2. **import module1,module2,module3**
  3. **importmodule1asm**
  4. **importmodule1as m1,module2asm2,module3**
  5. **frommoduleimportmember**
  6. **frommoduleimportmember1,member2,memebr3**
  7. **frommoduleimportmemeber1asx**
  8. **frommoduleimport\***



MemberAliasing:

1)fromdurgamathimportxas y,addassum

**2)print(y)**

3)sum(10,20)

Oncewedefinedas aliasname,weshouldusealiasnameonlyandweshouldnotuse original name

**1)from durgamathimportx asy**

**2)print(x)****NameError:name'x'is notdefined**

Reloadinga Module:

Bydefaultmodulewillbeloadedonlyonceeventhoughweareimportingmultiple multiple times.

**module1.py:**

print("Thisisfrom module1")

**test.py**

1)importmodule1

**4)importmodule1**

**2)importmodule1**

3)importmodule1

5)print("Thisistest module")

**Output**

Thisisfrommodule1 This is test module

* **Intheaboveprogramtestmodule willbeloadedonlyonceeventhoughweare importing multiple times.**
* **Theprobleminthisapproachisafterloadingamoduleifitisupdatedoutsidethen updated version of module1 is not available to our program.**
* **Wecan solvethisproblembyreloadingmoduleexplicitlybasedonour requirement.**
* **Wecan reloadbyusingreload()functionofimp module.**

**1)importimp**

**2)imp.reload(module1)**



**test.py:**

1)importmodule1

**6)reload(module1)**

**4)reload(module1)**

**2)importmodule1**

3)fromimpimportreload

5)reload(module1)

7)print("Thisistest module")

Intheaboveprogrammodule1willbeloaded4timesinthat1timebydefaultand3times explicitly. In this case output is

1)Thisisfrom module1

**4)Thisisfrom module1**

**2)Thisisfrom module1**

3)Thisisfrom module1

5)Thisistestmodule

Themainadvantageofexplicitmodulereloadingiswecanensurethatupdatedversionis always available to our program.

FindingMembersofModulebyusingdir()Function:

Pythonprovidesinbuiltfunctiondir()tolistoutallmembersofcurrentmoduleora specified module.

dir() To list out all members of current module dir(moduleName)Tolistoutallmembersofspecifiedmodule

**Eg1:test.py**

1)x=10

**4) print("Hello")**

**2)y=20**

3)deff1():

5)print(dir()) #Toprintallmembersofcurrent module

**Output**

['annotations', 'builtins', 'cached','doc', 'file', 'loader', 'nam e', 'package', 'spec', 'f1', 'x', 'y']

**166**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**Eg2:Todisplaymembersofparticularmodule durgamath.py:**

**1)x=888**

**3)defadd(a,b):**

**5)**

**7) print("TheProduct:",a\*b)**

**6)defproduct(a,b):**

**4) print("TheSum:",a+b)**

**2)**

**test.py:**

**1)importdurgamath**

**2)print(dir(durgamath))**

**Output**

['builtins','cached','doc','file','loader','name',

'package','spec', 'add', 'product', 'x']

Note:Foreverymoduleatthetimeofexecution Pythoninterpreterwilladdsomespecial properties automatically for internal use.

Eg: builtins, cached,'doc, file, loader,name, package,

spec

Basedonourrequirementwecanaccessthesepropertiesalsoinour program.

**Eg:test.py**

1)print(builtins )

|  |
| --- |
| **2)print(cached )** |
| **3)print(doc)** |
| **4)print(file)** |
| **5)print(loader)** |
| **6)print(name)** |
| **7)print(package)** |
| **8)print(spec)** |

**Output**

<module'builtins'(built-in)>None

None

**167**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



**test.py**

1)<\_frozen\_importlib\_external.SourceFileLoaderobjectat0x00572170>

**4)None**

**2)main**

3)None

**The Special Variable name :**

* **ForeveryPythonprogram, aspecialvariablename willbeadded internally.**
* **Thisvariablestoresinformationregardingwhethertheprogramisexecutedasan individual program or as a module.**
* **Iftheprogramexecutedasanindividualprogramthenthevalueofthisvariableis**

main

* **Iftheprogramexecutedasamodulefromsomeotherprogramthenthevalueofthis variable is the name of module where it is defined.**
* **Hence by using this name variablewecanidentifywhethertheprogramexecuted directly or as a module.**

**Demoprogram:**

**module1.py:**

1. **deff1():**

|  |
| --- |
| **2) ifname=='main':** |
| **3) print("Thecodeexecutedasaprogram")** |
| **4) else:** |
| **5) print("Thecodeexecutedasamodulefromsomeotherprogram")** |
| **6)f1()** |

**test.py:**

**1)importmodule1**

**2)module1.f1()**

D:\Python\_classes>pymodule1.py The code executed as a program

D:\Python\_classes>pytest.py

Thecodeexecutedasamodulefromsomeotherprogram Thecodeexecuted as amodulefromsomeotherprogram

**168**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



WorkingwithmathModule:

* **Pythonprovidesinbuiltmodulemath.**
* **Thismoduledefinesseveralfunctionswhichcanbeusedformathematicaloperations.**
* **Themainimportantfunctionsare**
  1. **sqrt(x)**
  2. **ceil(x)**
  3. **floor(x)**
  4. **fabs(x)**
  5. **log(x)**
  6. **sin(x)**
  7. **tan(x)**
  8. **....**

1)from mathimport\*

|  |
| --- |
| **2)print(sqrt(4))** |
| **3)print(ceil(10.1))** |
| **4)print(floor(10.1))** |
| **5)print(fabs(-10.6))** |
| **6)print(fabs(10.6))** |

**Output**

2.0

11

10

10.6

10.6

Note:Wecanfindhelpforanymodulebyusinghelp() function

**Eg:**

importmathhelp(math)

WorkingwithrandomModule:

* **Thismoduledefinesseveralfunctionstogeneraterandomnumbers.**
* **Wecanusethesefunctionswhiledevelopinggames,incryptographyandtogenerate random numbers on fly for authentication.**

1. **random() Function:**

Thisfunctionalwaysgeneratesomefloatvaluebetween0and1(notinclusive)

0<x<1



* 1. **fromrandomimport \***

3) print(random())

**Output**

**2)for iinrange(10):**

0.4572685609302056

0.6584325233197768

0.15444034016553587

0.18351427005232201

0.1330257265904884

0.9291139798071045

0.6586741197891783

0.8901649834019002

0.25540891083913053

0.7290504335962871

1. **randint()Function:**

Togeneraterandomintegerbeweentwogiven numbers(inclusive)

* 1. **fromrandomimport \***

**2)for iinrange(10):**

3) print(randint(1,100))#generaterandomintvaluebetween1and100(inclusive)

**Output**

51

44

39

70

49

74

52

10

40

8

1. **uniform() Function:**

Itreturnsrandomfloatvaluesbetween2givennumbers(notinclusive)

* 1. **fromrandomimport \***

**2)for iinrange(10):**

3) print(uniform(1,10))



**Output**

9.787695398230332

6.81102218793548

8.068672144377329

8.567976357239834

6.363511674803802

2.176137584071641

4.822867939432386

6.0801725149678445

7.508457735544763

1.9982221862917555

random() in between 0 and 1 (not inclusive) randint(x,y) in between x and y ( inclusive) uniform(x,y)inbetweenxandy(notinclusive)

1. **randrange([start],stop,[step])**

* **Returnsarandomnumberfromrange**
* **start<= x<stop**
* **startargumentisoptionalanddefaultvalueis0**
* **stepargumentisoptionalanddefaultvalueis1**
* **randrange(10)****generatesanumberfrom0to9**
* **randrange(1,11)****generatesanumberfrom1to 10**
* **randrange(1,11,2)****generatesanumberfrom1,3,5,7,9**
  1. **fromrandomimport \***

**2)for iinrange(10):**

3) print(randrange(10))

**Output:9**

4

0

2

9

4

8

9

5

9

1)fromrandomimport\*

**2)for iinrange(10):**

3) print(randrange(1,11))



**Output:2**

2

8

10

3

5

9

1

6

3

1)fromrandomimport\*

**2)for iinrange(10):**

3) print(randrange(1,11,2))

**Output:1**

3

9

5

7

1

1

1

7

3

1. **choice()Function:**

* **Itwon’treturnrandom number.**
* **Itwillreturn arandomobjectfromthegivenlistortuple.**

**1)fromrandomimport\***

**3)for iinrange(10):**

**print(choice(list))**

**4)**

**2)list=["Sunny","Bunny","Chinny","Vinny","pinny"]**

OutputBunny pinny Bunny Sunny Bunny pinny

pinny Vinny Bunny Sunny



## PACKAGES



֍Itisanencapsulationmechanismtogrouprelatedmodulesintoasingleunit.

֍packageisnothingbutfolderordirectorywhichrepresentscollectionof Python modules.

֍Anyfolderordirectorycontainsinit.pyfile,isconsideredasa Pythonpackage.This file can be empty.

֍A packagecancontainssubpackagesalso.

|  |  |  |
| --- | --- | --- |
| **x.py** |  | **y.py** |

Loan

**init.py**

**init.py**

**init.py**

**Home Loan**

**VehicleLoan**

**n.py**

**m.py**

**File1**

**File1**

**File1**

**init.py**

**init.py**

**init.py**

**Home Loan**

**VehicleLoan**

**Module1**

**Module1**

**Module1**

**Module1**

**File1**

**File1**

**File1**

Loan

Themainadvantagesofpackagestatementare

1. **Wecanresolvenamingconflicts**
2. **Wecanidentifyourcomponents uniquely**
3. **Itimprovesmodularityoftheapplication**



**Eg1:**

D:\Python\_classes>

|-test.py

|-pack1

|-module1.py

|-init.py

init.py:

emptyfile

**module1.py:**

def f1():

print("Hellothisisfrommodule1presentinpack1")

**test.py (version-1):importpack1.module1 pack1.module1.f1()**

**test.py(version-2):**

frompack1.module1importf1 f1()

**Eg2:**

D:\Python\_classes>

|-test.py

|-com

|-module1.py

|-init.py

|-durgasoft

|-module2.py

|-init.py

init.py:

emptyfile

**module1.py:**

def f1():

print("Hellothisisfrommodule1presentincom")

**module2.py:**

def f2():

print("Hellothisisfrommodule2presentincom.durgasoft")



**test.py**

**1)fromcom.module1importf1**

**4)f2()**

**2)fromcom.durgasoft.module2importf2**

**3)f1()**

**Output**

D:\Python\_classes>pytest.py

Hellothisisfrommodule1 presentincom

Hello thisisfrommodule2 presentincom.durgasoft

Note:Summarydiagramoflibrary, packages,moduleswhichcontainsfunctions,classes and variables.

Library

pack 1 pack 2 --------------- pack n

module1module2modulen module1module2modulen

function1function2functionnvariables classes



**177**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**

**100**

**PATTERN**

# PROGRAMS

Pattern-1:

**\*\*\*\*\*\*\*\*\*\***

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **print("\*"\*n)**

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

Pattern-2:

1111111111

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(i,end="")**
5. **print()**

2222222222

3333333333

4444444444

5555555555

6666666666

7777777777

88888888889999999999

101010 1010 1010 1010 10

Pattern-3:

12345678910

12345678910

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(j,end="")**
5. **print()**

12345678910

12345678910

12345678910

12345678910

12345678910

12345678910

12345678910

12345678910

**178**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



Pattern-4:

AAAAAAAAAA

B B B B BB B BB B

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(chr(64+i),end="")**
5. **print()**

C C C CC CC CC C

D D D D DD D DD D

EEEEEEEEEE

F F FF FF FF F F

GGGGGGGGGG

HHHHHHHHHH I I I I I I I I I I

J J J J J JJ J JJ

Pattern-5:

ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ ABCDEFGHIJ AB CD EF GH IJ

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(chr(64+j),end="")**
5. **print()**

Pattern-6:

101010 1010 1010 1010 10

9999999999

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(n+1-i,end="")**
5. **print()**

8888888888

7777777777

6666666666

5555555555

4444444444

3333333333

2222222222

1111111111



Pattern-7:

10987654321

10987654321

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(n+1-j,end="")**
5. **print()**

10987654321

10987654321

10987654321

10987654321

10987654321

10987654321

10987654321

10987654321

Pattern-8:

JJJJJJJJJJ I I I I I I I I I I

1. **n=int(input("Enterthenumberofrows:"))**
2. **for i inrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(chr(65+n-i),end="")**
5. **print()**

HHHHHHHHHH GGGGGGGGGG F F F F F F F F F F

EEEEEEEEEE

DDDDDDDDDD C C C C C C C C C C

B B B B B B B B B B AAAAAAAAAA

Pattern-9:

JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA JIHGFEDCBA J IHGFED CBA

1. **n=int(input("Enterthenumberofrows:"))**
2. **for i inrange(1,n+1):**
3. **for jin range(1,n+1):**
4. **print(chr(65+n-j),end="")**
5. **print()**



Pattern-10:

**Code-1**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

Pattern-11:

1

22

333

4444

55555

666666

7777777

88888888

999999999

101010 1010 1010 1010 10

Pattern-12:

**Code-2**

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,i+1):**
4. **print(i,end="")**
5. **print()**
6. **n=int(input("Enterthenumberofrows:"))**
7. **for iinrange(1,n+1):**
8. **for jin range(1,i+1):**
9. **print("\*",end="")**
10. **print()**
11. **n=int(input("Enterthenumberofrows:"))**
12. **for iinrange(1,n+1):**
13. **print("\*"\*i)**

1

12

123

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,i+1):**
4. **print(j,end="")**
5. **print()**

1234

12345

123456

1234567

12345678

123456789

12345678910



Pattern-13:

A

B B

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,i+1):**
4. **print(chr(64+i),end="")**
5. **print()**

C C C

D D D D

EEEEE

F F FF FF

GGGGGGG

HHHHHHHH I I I I I I I I I

J J J J J JJ J JJ

Pattern-14:

AAB

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **for jin range(1,i+1):**
4. **print(chr(64+j),end="")**
5. **print()**

AB C

A B C DABCDE

AB CD EF

AB CD EF G

A B C D E F G H AB CD EF GH I

AB CD EF GH IJ

1. **Squares**
2. **RightAngledTriangle**
3. **ReverseofRightAngledTriangle**

Pattern-15:

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print("\*",end="")**
5. **print()**

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\***

**\*\*\*\***

**\*\*\***

**\*\***

**\***



Pattern-16:

1111111111

222222222

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(i,end="")**
5. **print()**

33333333

4444444

555555

66666

7777

888

99

10

Pattern-17:

12345678910

123456789

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(j,end="")**
5. **print()**

12345678

1234567

123456

12345

1234

123

12

1

Pattern-18:

AAAAAAAAAA

B B B B BB B BB

C C C CC CC C

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(chr(64+i),end="")**
5. **print()**

D D D D DD D

EEEEEE

F F FF F

GGGG

HHH I I

J



Pattern-19:

ABCDEFGHIJ A B C D E F G H I A B C D E F G H

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(chr(64+j),end="")**
5. **print()**

ABCDEFG A B C D E F

ABCDE A B C DA B C

AB A

Pattern-20:

101010 1010 1010 1010 10

999999999

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(n+1-i,end="")**
5. **print()**

88888888

7777777

666666

55555

4444

333

22

1

Pattern-21:

10987654321

1098765432

109876543

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(n+1-j,end="")**
5. **print()**

10987654

1098765

109876

10987

1098

109

10



Pattern-22:

JJJJJJJJJJ I I I I I I I I I

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(chr(65+n-i),end="")**
5. **print()**

HHHHHHHH G G G G G G G

FFFFFF E E E E E D D D DC C C

BB A

Pattern-23:

JIHGFEDCBA J I H G F E D C B

JIHGFEDC J I H G F E D

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **forjinrange(1,n+2-i):**
4. **print(chr(65+n-j),end="")**
5. **print()**

JIHGFE J I H G F

JIHG J I H

JI J

Pattern-24:

**\***

**\*\***

**\*\*\***

**\*\*\*\***

1. **n=int(input("Enterthenumberofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),"\*"\*i,end="")**
4. **print()**

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**185**

**DURGASOFT,#202,2Floor,HUDAMaitrivanam,Ameerpet,Hyderabad-500038,**

**nd**



Pattern-25:

**\***

**\*\***

**1)n=int(input("Enterthenumberofrows:"))**

**\*\*\***

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jinrange(1,i+1):** |
| **5) print("\*",end="")** |
| **6) print()** |

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

Pattern-26:

1

22

333

4444

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),(str(i)+"")\*i)**
4. **print()**

55555

666666

7777777

88888888

999999999

10101010101010101010



Pattern-27:

1

**1)n=int(input("Enterthenumberofrows:"))**

12

123

|  |
| --- |
| **2)for i inrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i+1):** |
| **5) print(j,end="")** |
| **6) print()** |

1234

12345

123456

1234567

12345678

123456789

12345678910

Pattern-28:

A

B B

C CC

1. **n=int(input("Enterthenumberofrows:"))**
2. **for i inrange(1,n+1):**
3. **print(""\*(n-i),(chr(64+i)+"")\*i)**
4. **print()**

D DD D

EEEEE

F FF FF F

GGGGGGG

H HH H H HH H

Pattern-29:

AA BABC

**1)n=int(input("Enterthenumberofrows:"))**

A B C D A B C D EAB CD EF

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jinrange(1,i+1):** |
| **5) print(chr(64+j),end="")** |
| **6) print()** |

A B C D E F G A B C D E F G H AB C DEFGHI

AB CD EF GH IJ



Pattern-30:

**\*\*\*\*\***

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(i-1),"\*"\*(n+1-i))**

**\*\*\*\***

**\*\*\***

**\*\***

**\***

Pattern-31:

55555

1. **n=int(input("Enterthenumber ofrows:"))**
2. **foriinrange(1,n+1):**
3. **print(""\*(i-1),(str(n+1-i)+"")\*(n+1-i))**

4444

333

22

1

Pattern-32:

12345

**1)n=int(input("Enterthenumber ofrows:"))**

|  |
| --- |
| **2)foriinrange(1,n+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,n+2-i):** |
| **5) print(j,end="")** |
| **6) print()** |

1234

123

12

1

Pattern-33:

E E E E E DD D D

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(i-1),(str(chr(65+n-i))+"")\*(n+1-i))**

CCC BB

A

Pattern-34:

ABCDE ABCD A B C

**1)n=int(input("Enterthenumberofrows:"))**

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(65,66+n-i):** |
| **5) print(chr(j),end="")** |
| **6) print()** |

AB

A



Pattern-35:

**\***

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),"\*"\*(2\*i-1))**

**\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

Pattern-36:

1

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),(str(i)+"")\*(2\*i-1))**

222

33333

4444444

555555555

Pattern-37:

A

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),(str(chr(64+i)+""))\*(2\*i-1))**

B B B

C CC CC

DD D DD D D

EEEEEEEEE

Pattern-38:

ACCC

1. **n=int(input("Enterthenumber ofrows:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),(str(chr(64+2\*i-1)+""))\*(2\*i-1))**

EEEEE

GGGGGGG I I I I I I I I I

Pattern-39:

**1)n=int(input("Enterthenumberofrows:"))**

1

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jinrange(1,2\*i):** |
| **5) print(j,end="")** |
| **6) print()** |

123

12345

1234567

123456789



Pattern-40:

1

**1)n=int(input("Enterthenumberofrows:"))**

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) forjinrange(2\*i-1,0,-1):** |
| **5) print(j,end="")** |
| **6) print()** |

321

54321

7654321

987654321

Pattern-41:

**1)n=int(input("Enterthenumberofrows:"))**

A

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) forjinrange(65,65+2\*i-1):** |
| **5) print(chr(j),end="")** |
| **6) print()** |

ABC

ABC DE

AB C DEFG

AB CD EF GH I

Pattern-42:

**1)n=int(input("Enterthenumber ofrows:"))**

A

|  |
| --- |
| **2)for i inrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) forjinrange(65+2\*i-2,64,-1):** |
| **5) print(chr(j),end="")** |
| 6)**print()** |

CBA

EDC BA

GF ED C BA

IH GF ED CB A

Pattern-43:

**1)n=int(input("Enterthenumber ofrows:"))**

0

|  |
| --- |
| **2)for i inrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i):** |
| **5) print(i-j,end="")** |
| **6) forkinrange(0,i):** |
| **7) print(k,end="")** |
| 8)**print()** |

101

21012

3210123

432101234



Pattern-44:

A

**1)n=int(input("Enterthenumberofrows:"))**

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i):** |
| **5) print(chr(i-j+65),end="")** |
| **6) forkinrange(0,i):** |
| **7) print(chr(k+65),end="")** |
| **8) print()** |

BAB

C BABC

D CB AB C D

ED C BAB C DE

Pattern-45:

**1)n=int(input("Enterthenumberofrows:"))**

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i+1):** |
| **5) print(j,end="")** |
| **6) forkinrange(i-1,0,-1):** |
| **7) print(k,end="")** |
| **8) print()** |

1

121

12321

1234321

123454321

Pattern-46:

**1)n=int(input("Enterthenumberofrows:"))**

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i+1):** |
| **5) print(chr(64+j),end="")** |
| **6) forkinrange(1,i):** |
| **7) print(chr(64+k),end="")** |
| **8) print()** |

A

ABA

ABCAB

AB C DABC

AB CD EABC D

Pattern-47:

|  |
| --- |
| **1)n=int(input("Enteranumber:"))** |

5

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) print(""\*(n-i),end="")** |
| **4) for jin range(1,i+1):** |
| **5) print(n+1-j,end="")** |
| **6) print()** |

54

543

5432

54321



Pattern-48:

**\*\*\*\*\*\*\*\*\***

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print("\*",end="")** |
| **6) forkinrange(1,num+1-i):** |
| **7) print("\*",end="")** |
| **8) print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

**\*\*\*\*\*\*\***

**\*\*\*\*\***

**\*\*\***

**\***

Pattern-49:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(0,num+1-i):** |
| **5) print(num+1-i,end="")** |
| **6) forkinrange(1,num+1-i):** |
| **7) print(num+1-i,end="")** |
| **8) print()** |

555555555

4444444

33333

222

1

Pattern-50:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(0,num+1-i):** |
| **5) print(2\*num+1-2\*i,end="")** |
| **6) forkinrange(1,num+1-i):** |
| **7) print(2\*num+1-2\*i,end="")** |
| **8) print()** |

999999999

7777777

55555

333

1

Pattern-51:

**1)num=int(input("Enteranumber:"))**

1234567

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(j,end="")** |
| **6) forkinrange(2,num+2-i):** |
| **7) print(num+k-i,end="")** |
| 8)**print()** |

12345

123

1



Pattern-52:

EEEEEEEEE DDDDDDD C C C C C

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(65+num-i),end="")** |
| **6) forkinrange(2,num+2-i):** |
| **7) print(chr(65+num-i),end="")** |
| 8)**print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

BBB A

Pattern-53:

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(65+2\*num-2\*i),end="")** |
| **6) forkinrange(2,num+2-i):** |
| **7) print(chr(65+2\*num-2\*i),end="")** |
| 8)**print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

IIIIIIIII

GGGGGGGEEEEE

CCC A

Pattern-54:

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(64+j),end="")** |
| **6) forkinrange(2,num+2-i):** |
| **7) print(chr(68+k-i),end="")** |
| **8) print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

ABCDEFG A B C D E

ABC A

Pattern-55:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(j,end="")**
6. **print()**
7. **for kinrange(1,num):**
8. **print(""\*k,end="")**
9. **forlinrange(1,num+1-k):**
10. **print(l,end="")**
11. **print()**

1

12

123

1234

12345

1234

123

12

1



Pattern-56:

**\***

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print("\*",end="")**
6. **print()**
7. **forkin range(1,num):**
8. **print(""\*k,end="")**
9. **forlinrange(1,num+1-k):**
10. **print("\*",end="")**
11. **print()**

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\***

**\*\*\***

**\*\***

**\***

Pattern-57:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(num-j,end="")**
6. **print()**
7. **forkin range(1,num):**
8. **print(""\*k,end="")**
9. **forlinrange(1,num+1-k):**
10. **print(num-l,end="")**
11. **print()**

4

43

432

4321

43210

4321

432

43

4

Pattern-58:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(0,i):**
5. **print(num+j-i,end="")**
6. **print()**
7. **forkin range(1,num):**
8. **print(""\*k,end="")**
9. **forlinrange(1,num+1-k):**
10. **print(l+k-1,end="")**
11. **print()**

3

23

123

0123

123

23

3



Pattern-59:

E D E

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jinrange(0,i):**
5. **print(chr(65+num+j-i),end="")**
6. **print()**
7. **for kinrange(1,num):**
8. **print(""\*k,end="")**
9. **forlinrange(0,num-k):**
10. **print(chr(65+k+l),end="")**
11. **print()**

C D EBC D E

ABC D E

BC D E

C DE

DE E

Pattern-60:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **for jinrange(1,i+1):**
4. **print("\*",end="")**
5. **print()**
6. **forainrange(1,num+1):**
7. **forkinrange(1,num+1-a):**
8. **print("\*",end="")**
9. **print()**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\***

**\*\*\***

**\*\***

**\***

Pattern-61:

4

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **for jinrange(1,i+1):**
4. **print(num-j,end="")**
5. **print()**
6. **forainrange(1,num+1):**
7. **forkinrange(1,num+1-a):**
8. **print(num-k,end="")**
9. **print()**

43

432

4321

43210

4321

432

43

4



Pattern-62:

4

1. **num=int(input("Enteranumber:"))**
2. **foriinrange(1,num+1):**
3. **for jin range(1,i+1):**
4. **print(num-i+j-1,end="")**
5. **print()**
6. **forainrange(1,num+1):**
7. **for kinrange(0,num-a):**
8. **print(k+a,end="")**
9. **print()**

34

234

1234

01234

1234

234

34

4

Pattern-63:

E

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **for jin range(1,i+1):**
4. **print(chr(65+num-i),end="")**
5. **print()**
6. **forainrange(1,num+1):**
7. **for kinrange(0,num-a):**
8. **print(chr(65+a),end="")**
9. **print()**

D D C C C

B B B B

AAAAA

B B B B

C C C

DD E

Pattern-64:

E

1. **for iinrange(1,num+1):**
2. **for jin range(1,i+1):**
3. **print(chr(65+num-j),end="")**
4. **print()**
5. **forainrange(1,num+1):**
6. **forkinrange(num-a,0,-1):**
7. **print(chr(64+k+a),end="")**
8. **print()**
9. **num=int(input("Enteranumber:"))**

ED

ED C

ED C B

EDCBA E D C B

EDC E D

E



Pattern-65:

E

1. **num=int(input("Enteranumber:"))**
2. **foriinrange(1,num+1):**
3. **for jin range(1,i+1):**
4. **print(chr(64+num-i+j),end="")**
5. **print()**
6. **forainrange(1,num+1):**
7. **for kinrange(1,num-a+1):**
8. **print(chr(64+k+a),end="")**
9. **print()**

D E

C D E

B C D E

AB CD E

B C D E

C D E

DE E

Pattern-66:

**1)num=int(input("Enteranumber:"))**

**\***

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jinrange(1,i+1):** |
| **5) print("\*",end="")** |
| 6)**print()** |

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

Pattern-67:

**1)num=int(input("Enteranumber:"))**

1

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jinrange(1,i+1):** |
| **5) print(i,end="")** |
| 6)**print()** |

22

333

4444

55555

Pattern-68:

1

**1)num=int(input("Enteranumber:"))**

12

|  |
| --- |
| **2)foriinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(1,1+i):** |
| **5) print(j,end="")** |
| 6)**print()** |

123

1234

12345



Pattern-69:

A

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jinrange(1,1+i):** |
| **5) print(chr(64+i),end="")** |
| **6) print()** |

BB

C CC

D D DD

EEEEE

Pattern-70:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jinrange(1,1+i):** |
| **5) print(chr(64+j),end="")** |
| **6) print()** |

AA BAB C

A B C D ABC DE

Pattern-71:

**1)num=int(input("Enteranumber:"))**

**\*\*\*\*\***

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print("\*",end="")** |
| 6)**print()** |

**\*\*\*\***

**\*\*\***

**\*\***

**\***

Pattern-72:

**1)num=int(input("Enteranumber:"))**

55555

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(num-i+1,end="")** |
| 6)**print()** |

4444

333

22

1

Pattern-73:

**1)num=int(input("Enteranumber:"))**

54321

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(num+2-i-j,end="")** |
| 6)**print()** |

4321

321

21

1



Pattern-74:

EEEEE DDDD C C C

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(65+num-i),end="")** |
| 6)**print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

BB A

Pattern-75:

**1)num=int(input("Enteranumber:"))**

EDCBA D C B A C B A

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(65+num+1-i-j),end="")** |
| 6)**print()** |

BA A

Pattern-76:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(1,num+2-i):** |
| **5) print(chr(64+j),end="")** |
| **6) print()** |

ABCDE A B C D A B C

AB A

Pattern-77:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jinrange(1,i+1):**
5. **print("\*",end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print("\*",end="")**
11. **print()**

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\***

**\*\*\***

**\*\***

**\***



Pattern-78:

1

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(i,end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print(num-p,end="")**
11. **print()**

22

333

4444

55555

4444

333

22

1

Pattern-79:

1. **num=int(input("Enteranumber:"))**
2. **for i inrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(j,end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print(q+p,end="")**
11. **print()**

1

12

123

1234

12345

2345

345

45

5

Pattern-80:

1. **num=int(input("Enteranumber:"))**
2. **for i inrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(j,end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print(q,end="")**
11. **print()**

1

12

123

1234

12345

1234

123

12

1



Pattern-81:

A

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(chr(64+i),end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print(chr(64+num-p),end="")**
11. **print()**

BB

C CC

D D DD

E E E E E DDDD C C C

BB A

Pattern-82:

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jin range(1,i+1):**
5. **print(chr(64+j),end="")**
6. **print()**
7. **for pinrange(1,num):**
8. **print(""\*p,end="")**
9. **forqinrange(1,num+1-p):**
10. **print(chr(64+q+p),end="")**
11. **print()**

AA BABC

AB C D

AB CD E

B CD E

C DE

DE E

1. **n=int(input("Entera number:"))**
2. **for iinrange(1,n+1):**
3. **print(""\*(n-i),end="")**
4. **for jin range(1,i+1):**
5. **print(n-i+j,end="")**
6. **forkinrange(2,i+1):**
7. **print(n+1-k,end="")**
8. **print()**
9. **for iinrange(1,n+1):**
10. **print(""\*i,end="")**
11. **for jin range(1+i,n+1):**
12. **print(j,end="")**
13. **for kinrange(2,n+1-i):**
14. **print(n+1-k,end="")**
15. **print()**

Pattern-83:

5

454

34543

2345432

123454321

2345432

34543

454

5



Pattern-84:

5

**1)while True:**

|  |
| --- |
| **2) n=int(input("Enteranumber:"))** |
| **3) for iin range(1,n+1):** |
| **4) print(""\*(n-i),end="")** |
| **5) forjinrange(1,i+1):** |
| **6) print(n+1-j,end="")** |
| **7) forkinrange(2,i+1):** |
| **8) print(n-i+k,end="")** |
| **9) print()** |
| **10) for iin range(1,n+1):** |
| **11) print(""\*i,end="")** |
| **12) forjinrange(1,n+1-i):** |
| **13) print(n+1-j,end="")** |
| **14) forkinrange(2,n+1-i):** |
| **15) print(i+k,end="")** |
| 16) **print()** |

545

54345

5432345

543212345

5432345

54345

545

5

Pattern-85:

**\***

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print("\*",end="")** |
| **6) ifi>=2:** |
| **7) print(""\*(2\*i-4),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print("\*",end="")** |
| 10) **print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

**\*\***

**\*\***

**\* \***

**\* \***

Pattern-86:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(i,end="")** |
| **6) ifi>=2:** |
| **7) print(""\*(2\*i-4),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(i,end="")** |
| **10) print()** |

1

22

33

4 4

5 5



Pattern-87:

5

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(num+1-i,end="")** |
| **6) ifi>=2:** |
| **7) print(""\*(2\*i-4),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(num+1-i,end="")** |
| 10) **print()** |

44

33

2 2

1 1

Pattern-88:

E D D CC

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(chr(64+num+1-i),end="")** |
| **6) ifi>=2:** |
| **7) print(""\*(2\*i-4),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(chr(64+num+1-i),end="")** |
| 10) **print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

B B

A A

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

Pattern-89:

A

|  |
| --- |
| **2)for i inrange(1,num+1):** |
| **3) print(""\*(num-i),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(chr(64+i),end="")** |
| **6) ifi>=2:** |
| **7) print(""\*(2\*i-4),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(chr(64+i),end="")** |
| **10) print()** |

BB

CC

D D

E E



Pattern-90:

**\* \***

|  |
| --- |
| **2)for i inrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print("\*",end="")** |
| **6) ifi<=4:** |
| **7) print(""\*(2\*num-2\*i-2),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print("\*",end="")** |
| **10) print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

**\* \***

**\*\***

**\*\***

**\***

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

Pattern-91:

1 1

|  |
| --- |
| **2)for i inrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(i,end="")** |
| **6) ifi<num:** |
| **7) print(""\*(2\*num-2\*i-2),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(i,end="")** |
| **10) print()** |

2 2

33

44

5

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

Pattern-92:

5 5

|  |
| --- |
| **2)for i inrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) for jin range(i,i+1):** |
| **5) print(num-i+1,end="")** |
| **6) ifi<=4:** |
| **7) print(""\*(2\*num-2\*i-2),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(num-i+1,end="")** |
| **10) print()** |

4 4

33

22

1

Pattern-93:

**1)num=int(input("Enteranumber:"))**

|  |
| --- |
| **2)foriinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) forjinrange(i,i+1):** |
| **5) print(chr(64+num-i+1),end="")** |
| **6) ifi<=4:** |
| **7) print(""\*(2\*num-2\*i-2),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(chr(64+num-i+1),end="")** |
| **10) print()** |

EE

DD

C C B B A



Pattern-94:

A A

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) for jinrange(i,i+1):** |
| **5) print(chr(64+i),end="")** |
| **6) ifi<=4:** |
| **7) print(""\*(2\*num-2\*i-2),end="")** |
| **8) forkinrange(i,i+1):** |
| **9) print(chr(64+i),end="")** |
| **10) print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber:"))** |

B B

CC

DD E

Pattern-95:

**1)num=int(input("Enteranumber:"))**

**\*\*\*\*\***

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(i-1),end="")** |
| **4) for jinrange(1,num+1):** |
| **5) print("\*",end="")** |
| **6) print()** |

**\*\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\***

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(num-i),end="")**
4. **for jinrange(1,i+1):**
5. **print("\*",end="")**
6. **print(""\*(num-i),end="")**
7. **forkinrange(1,i+1):**
8. **print("\*",end="")**
9. **print()**

Pattern-96:

**\* \***

**\*\* \*\***

**\*\*\* \*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

Pattern-97:

**1)n=int(input("Enteranumber:"))**

1

|  |
| --- |
| **2)for iinrange(1,n+1):** |
| **3) for jinrange(1,i+1):** |
| **4) if(i%2!=0andj%2!=0)or(i%2==0andj%2==0):** |
| **5) print("1",end="")** |
| **6) else:** |
| **7) print("0",end="")** |
| 8)**print()** |

01

101

0101

10101

010101

1010101



Pattern-98:

**\***

1. **num=int(input("Enteranumber:"))**
2. **for iinrange(1,num+1):**
3. **print(""\*(2\*num-i+3),end="")**
4. **for jinrange(1,i+1):**
5. **print("\*",end="")**
6. **print()**
7. **for iinrange(1,num+3):**
8. **print(""\*(2\*num-i+1),end="")**
9. **forjin range(-1,i+1):**
10. **print("\*",end="")**
11. **print()**
12. **for iinrange(1,num+5):**
13. **print(""\*(2\*num-i),end="")**
14. **forjin range(-2,i+1):**
15. **print("\*",end="")**
16. **print()**
17. **for iinrange(1,num+3):**
18. **print(""\*((2\*num)),end="")**
19. **print("\*"\*3)**

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\*\***

**\*\*\***

**\*\*\***

**\*\*\***

**\*\*\***

**\*\*\***

**\*\*\***

Pattern-99:

|  |
| --- |
| **2)for iinrange(1,num+1):** |
| **3) print(""\*(2\*num-i),end="")** |
| **4) for jin range(1,i+1):** |
| **5) print("\*",end="")** |
| **6) print()** |
| **7)for iinrange(1,num+1):** |
| **8) print(""\*(num-i),end="")** |
| **9) for jin range(1,i+1):** |
| **10) print("\*",end="")** |
| **11) print(""\*(num-i),end="")** |
| **12) forkinrange(1,i+1):** |
| **13) print("\*",end="")** |
| **14) print()** |

|  |
| --- |
| **1)num=int(input("Enteranumber"))** |

**\***

**\*\***

**\*\*\***

**\*\*\*\***

**\*\*\*\*\***

**\* \***

**\*\* \*\***

**\*\*\* \*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***



Pattern-100:

**\*\***

1. **n=int(input("Entera number"))**
2. **for iinrange(1,2\*n+1):**
3. **ifi%2==0:**
4. **print("\*"\*i,end="")**
5. **else:**
6. **print("\*"\*(i+1),end="")**
7. **print()**

**\*\***

**\*\*\*\***

**\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

**\*\*\*\*\*\*\*\*\*\***

Pattern-101:

1)n=int(input("Enteranumber:"))

|  |
| --- |
| **2)forainrange(1,n+1,2):** |
| **3) for iin range(1,n+1):** |
| **4) print(""\*(2\*n-i-a),end="")** |
| **5) forjinrange(1,i+a):** |
| **6) print("\*",end="")** |
| **7) print()** |
| **8)for binrange(1,n+1):** |
| **9) print(""\*(n-2),end="")** |
| **10) print("\*"\*3)** |



|  |  |  |  |
| --- | --- | --- | --- |
|  | | **\*** |  |
| **\*\***  **\*\*\***  **\*\*\*\***  **\*\*\*\*\***  **\*\*\*\*\*\***  **\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\***  **\*\*\***  **\*\*\*\***  **\*\*\*\*\***  **\*\*\*\*\*\***  **\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*** |
| **\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\***  **\*\*\*\*\*\***  **\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\* \***  **\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\*** |
| **\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\*\*\*\*\*\*\*\*\*\*\*\***  **\*\*\***  **\*\*\***  **\*\*\***  **\*\*\***  **\*\*\***  **\*\*\***  **\*\*\***  **\*\*\*** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |