**Introduction to Python:**

Python is a high-level, versatile, and easy-to-learn programming language.

It emphasizes code readability and reduces the cost of program maintenance.

Python is widely used for web development, data analysis, machine learning, scientific computing, and more.

**Basic Concepts:**

**Variables and Data Types:**

Variables store data. They can hold different types of values, like numbers, strings, and lists.

Common data types: int, float, str (string), bool (boolean), list, tuple, dict (dictionary).

**Control Structures:**

Conditional Statements:

if, elif (else if), else.

Loops:

for loop (iterating over sequences), while loop (repeating code as long as a condition is true).

Functions:

Functions are blocks of reusable code. They help in modularity and reusability.

Functions can take parameters (inputs) and return values (outputs).

Lists and Dictionaries:

Lists: Ordered collections of items. Access elements by their index.

Dictionaries: Key-value pairs. Access values using keys.

Intermediate Concepts:

Object-Oriented Programming (OOP):

Python supports OOP principles: encapsulation, inheritance, and polymorphism.

Classes are templates for creating objects, while objects are instances of classes.

Exception Handling:

Handling errors and exceptions using try, except, else, and finally blocks.

File Handling:

Reading and writing to files. Useful for data storage and manipulation.

Advanced Concepts:

**Modules and Libraries:**

Modules are reusable files of code. Libraries provide additional functionalities.

Popular libraries: NumPy (numerical computing), pandas (data manipulation), matplotlib (plotting), and more.

**List Comprehensions:**

A concise way to create lists. It combines loops and conditions in a single line.

**Generators:**

Special type of iterable. They generate values on-the-fly, conserving memory.

Getting Started:

**Python:**

Python was created by Guido van Rossum, a Dutch programmer, in the late 1980s.

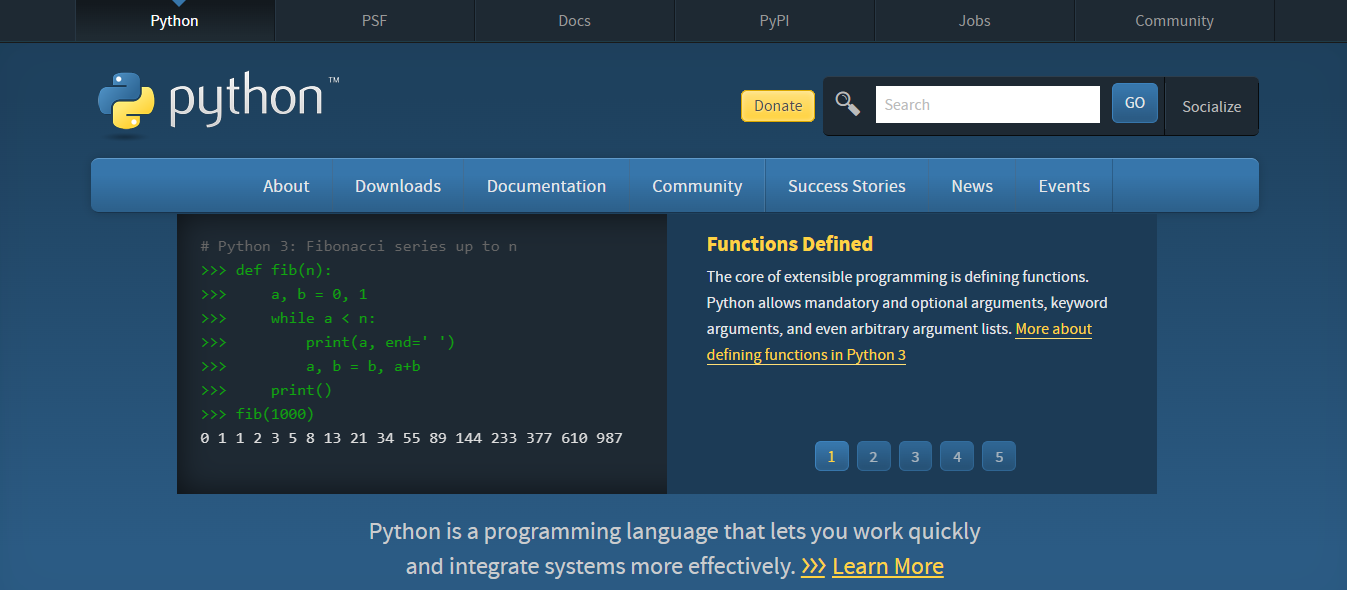
He started working on Python in December 1989 during his holidays in the Netherlands. He aimed to design a programming language that needed more code readability and wanted a simpler language that reduced the complexities.



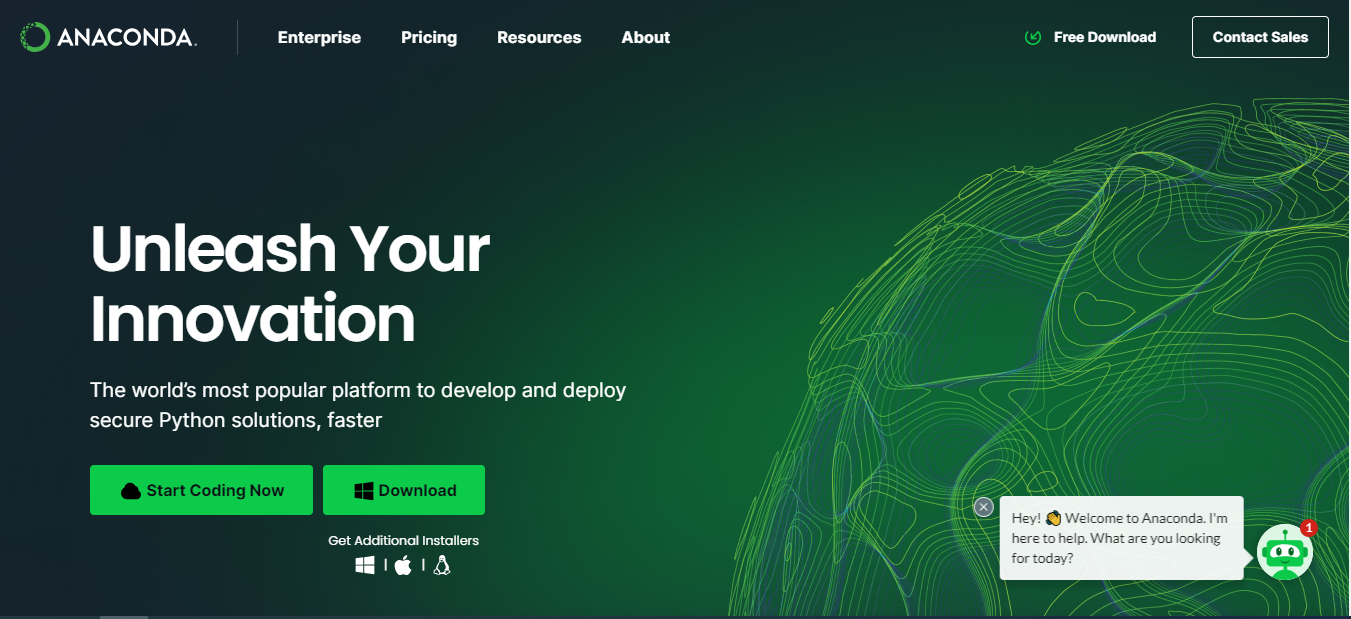
**(Image taken from the Wikipedia)**

**Installation:**

Download and install Python from the official website (python.org)

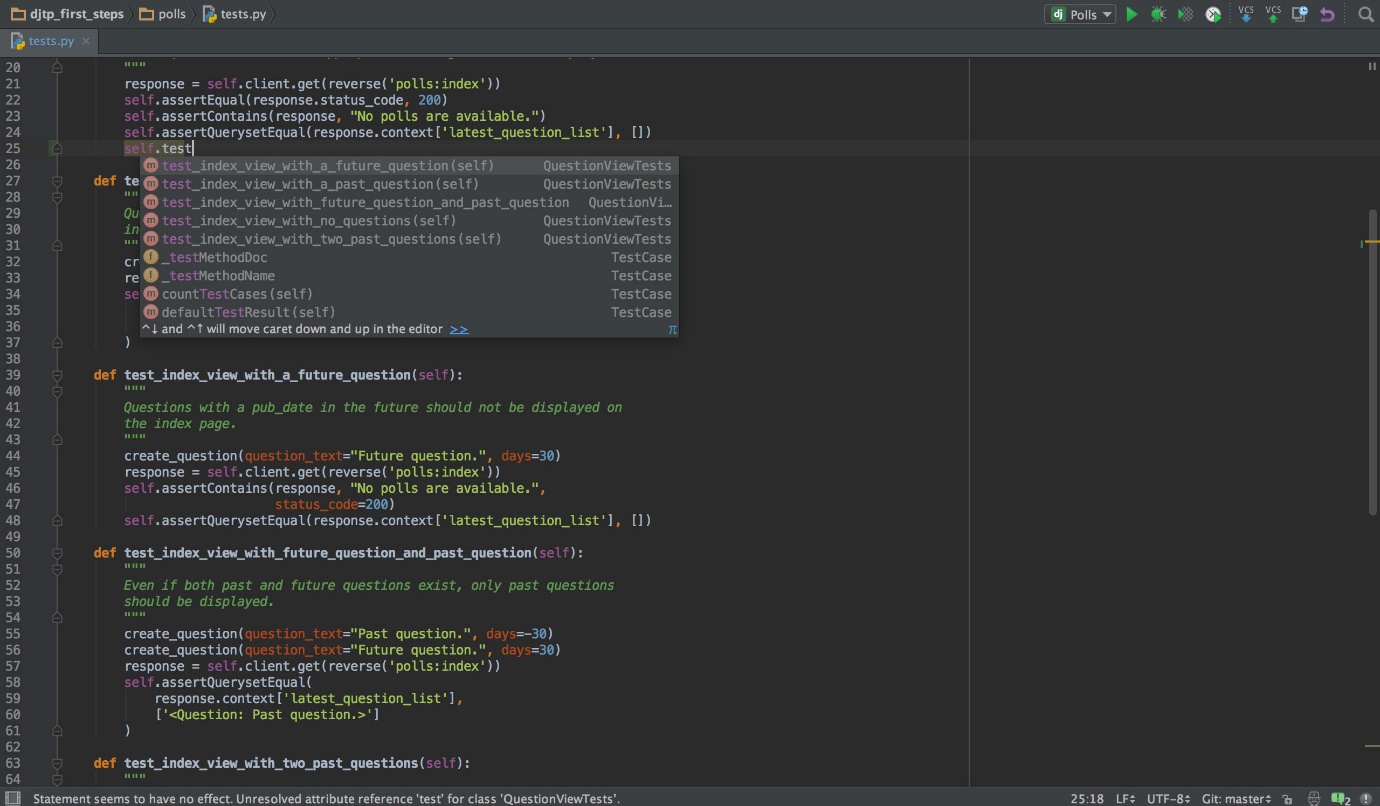


**Anaconda** <https://www.anaconda.com/>



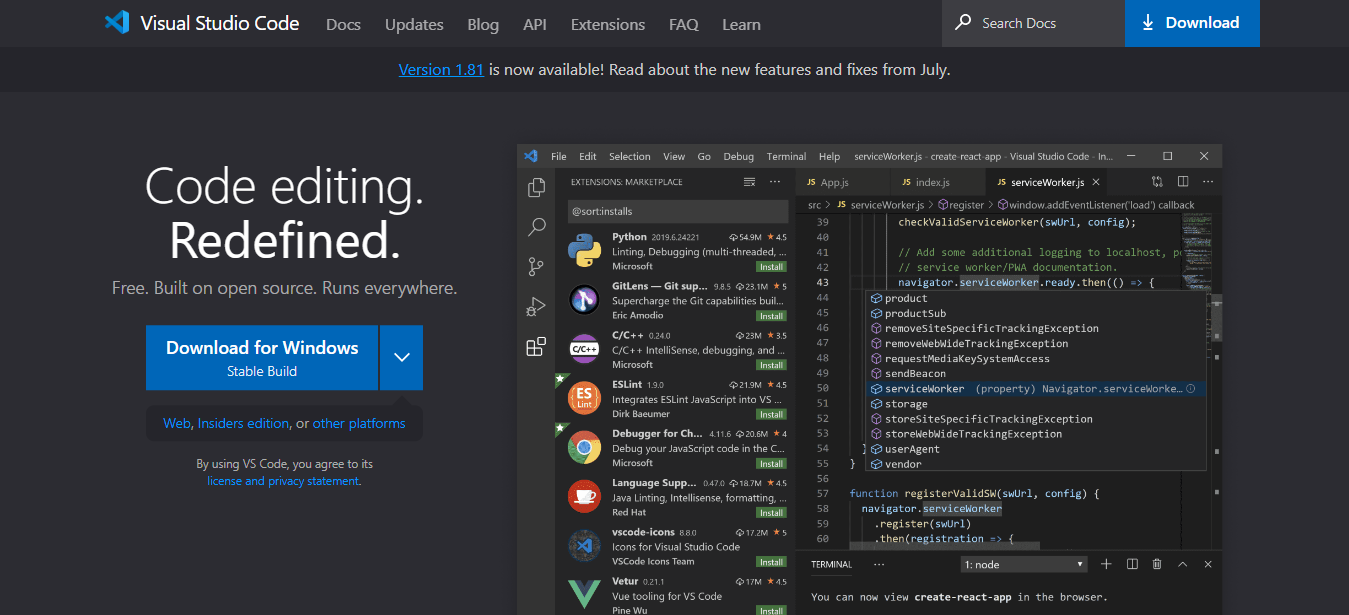
**Integrated Development Environments (IDEs):**

IDEs like PyCharm, Visual Studio Code, or Jupyter Notebook can be used to do code development and debugging.



Visual Studio Code:

<https://code.visualstudio.com/>



Always refer to the latest Python documentation and resources for the most accurate and up-to-date information.

**Variables and Data Types in Python:**

**\* Use of variables in python :**  Variables act as a container to hold the value or data structure

**\* Python is a dynamically typed programming language**

**\* Dynamically Typed programming language is one where reassigment of values for different datatypes is possible**

Variables and data types are fundamental concepts in programming. A variable is like a container, holding different types of information, such as numbers, words, or even heterogeneous values

Different Categories of data types in Python are:

1. Integers (int): Whole numbers like 2,3,4,5,23.
2. Floating-Point Numbers (float): Numbers with decimal points, like 3.14 or 1.4.
3. Strings (str): Sequence of characters, like "Hello, World!" or 'Welcome to Python!'.
4. Boolean (bool): True or False values.

Data Structures:

1. List: Ordered sequence of values
2. Sets: Unique set of values
3. Dictionary: key value pairs.

**Operation using int:**

**int operations –**

1. Addition(+) : Adds two integers

**Example:**

>>> sumNum = 5 + 3

sumNum is the variable name and 8 has been assigned to the variable sumNum

1. Subtraction(-): subtract one number from another

**Example:**

>>> subNum = 10 – 3

1. Multiplication(\*): Multiplies two integers

**Example:**

>>> mulNum = 5\*6

1. Division(/) : Divides one integer by another, yielding a floating point result. This gives us the quotient of the division of two numbers.

**Example**:

>>> divNum = 24/4

1. Modulus(%): The remainder of the division Is returned as the result when the following is applied

**Example:**

>>> modNum = 18%5

1. Exponentiation(\*\*): Integer to the power of another

**Example:**

>>> expNum = 2\*\*3

**Comparison Operations –**

1. Equals Operation (==) – If two integers are equal

**Example:**

>>> x = 6

>>> y = 6

>>> print(x == y)

1. Not equal (!=) – Check if two integers are not equal

**Example:**

>>> a = 10

>>> b = 20

>>> print(a != b)

1. Greater than (>) : Checks if one integer is greater than another

**Example:**

>>> c = 20

>>> d =10

>>> print(c > d)

1. Lesser than (<) : Checks if one integer is less than or equal to another

**Example:**

>>> e = 30

>>> f = 34

>>> print (e < f)

1. Absolute value (abs()) : Returns the absolute value of the integer

**Example:**

>>> g = abs(-10)

>>> print(g)

1. int() : Converts the value to an integer

**Example:**

>>> h = int(3.14)

>>> print(h)

1. round() : Rounds a floating-point number to the nearest integer.

**Example:**

k = round(3.7)

print(k)

1. **print String Formatting**

String formatting is done using the curly braces and the format function.

In the example below we shall use only the curly braces and not the indexing in the

**Example1:**

**Code**

>>> name=‘Rajesh’

>>> college = ‘SSDC’

>>> print(‘{} is the name of the student and the college name is {}’.format(name,college)

Rajesh is the name of the student and the college name is SSDC.

**Example2**

**Code**

**>>>** name = ‘Rajesh’

>>> college = ‘SSDC’

>>> print(‘{0} is the name of the student and the college name is {1}’.format(name,college)

Rajesh is the name of the student and the college name is SSDC.

**Integer and Float In-built functions**

**int():**

int() function converts a value to an integer data type.

**Syntax :** int(string,baseNumber)

**Code:**

#int(value, base=10)

>>> val1 = ‘42’

>>> val1 = int(val1) # The val1 value now is of type integer.

>>> binaryVal = ‘01010’

>>> val2 = int(binaryVal, base=2)

>>> print(val2)

**float():**

float() function converts a value to a floating data type.

**Example Code**

>>> Val2 = ‘4.14’

>>> Val2 = float(Val2)

**Capturing input from the command line - input():**

To capture input from the command line we can use the input() function.

* Input function always captures the input from the command line as string.
* When input is entered as integer or floating value the input is captured as string. To convert string (‘3’, ‘4’,’5’ ,‘6’) to int use int(‘4’). To onvert string(’10.5’,’8.4’) to float use float(’10.5’)

***Syntax* :** input(stringvalue)

***Example Code:***

>>> name = input(‘Enter your name ’)

Enter your name : Abhilash

>>> print(name)

Abhilash

**Functions:**

1. Functions are blocks of code that perform a certain behaviour or functionality.
2. Functions can be reused later
3. Functions can take input and return an output.
4. Functions help in writing modularized code.
5. Functions can also two types of arguments – positional arguments and default arguments.

***Syntax of function:***

def functionName(parameters):

<body>

return ouput

Notes:

def is the keyword followed by the function name and then parameters could be passed within the parenthisis

***Example Code:***

>>>def computeSum(a,b):

sumNum = a + b

return sumNum

>>> computeSum(10,20)

Output 30

**Notes:**

**Function name :** computeSum

**Parameters:** a,b

**Return type:** sumNum

**Default Arguments**

***Example2 Code:***

def printDetails (name, course=’N/A’):

print(‘Student name is {} and course name is {}’.format(name,course))

>>> printDetails(‘Rajesh’)

Output – Student name is Rajesh and course name is N/A

>>> printDetails(‘Rajesh’, ‘BBA’)

Output – Student name is Rajesh and course name is BBA

**Conditional Statements (if , elif and else)**

The conditional statements are decision-making statements and are written using the if, elif and else statements.

***Syntax:***

If condition1:

<Block of Code>

elif condition2:

<Block of Code>

elif condition3:

<Block of Code>

:

else:

***Example Code***

score = 90

if score < 40 :

print(‘Grade is D’)

elif score >= 40 and score < 50:

print(‘Grade is C’)

elif score >= 50 and score < 70:

print(‘Grade is B’)

else:

print(‘Grade is A)

**while Statement**

“while” statement with a condition can be used for iterating over multiple statements . The multiple statements can be used under “while” block.

***Syntax:***

while condition:

statements

***Example Code***

x = 5

while x < 10:

print(x)

x = x + 1

Keywords or reserve words in python:

1. False 2.class 3. finally 4.is 5.return 6. None 7. continue 8.for

9. lambda 10.try 11.True 12.def 13.from 14.nonlocal 15. while 16.and

17. del 18.global 19.not 20.with 21.as 22.elif 23.if 24.or 25. yield

26. assert 27. else 28. import 29. pass 30. break 31. except 32. in 33. raise