PYTHON PROGRAMMING

NUPAT DevOps Pathway Week 3

Lesson Objectives

At the end of this lesson, students should be able to:

1. Understand basics of programming.

2. Differentiate between interpreted and compiled languages.

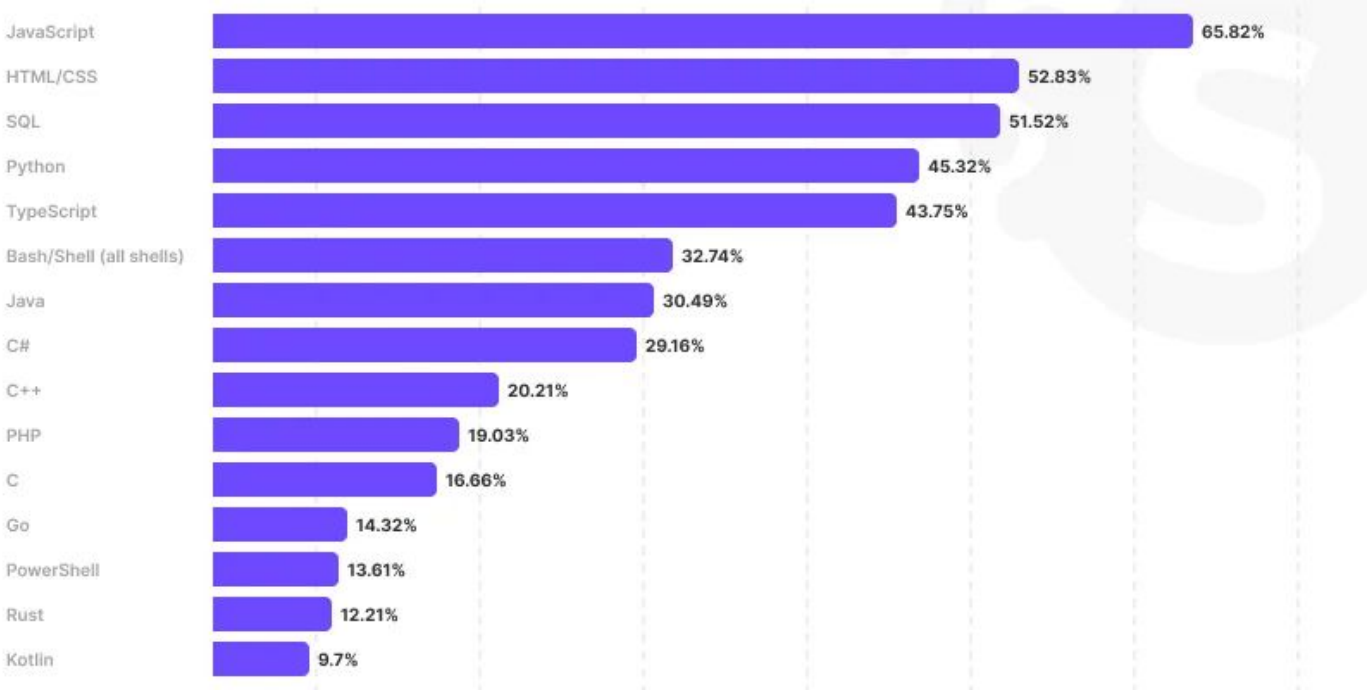
3. Write and execute basic programs.

4. Understand basics of Python programming.

5. Demonstrate python scripting proficiency in conditional statements and custom functions.

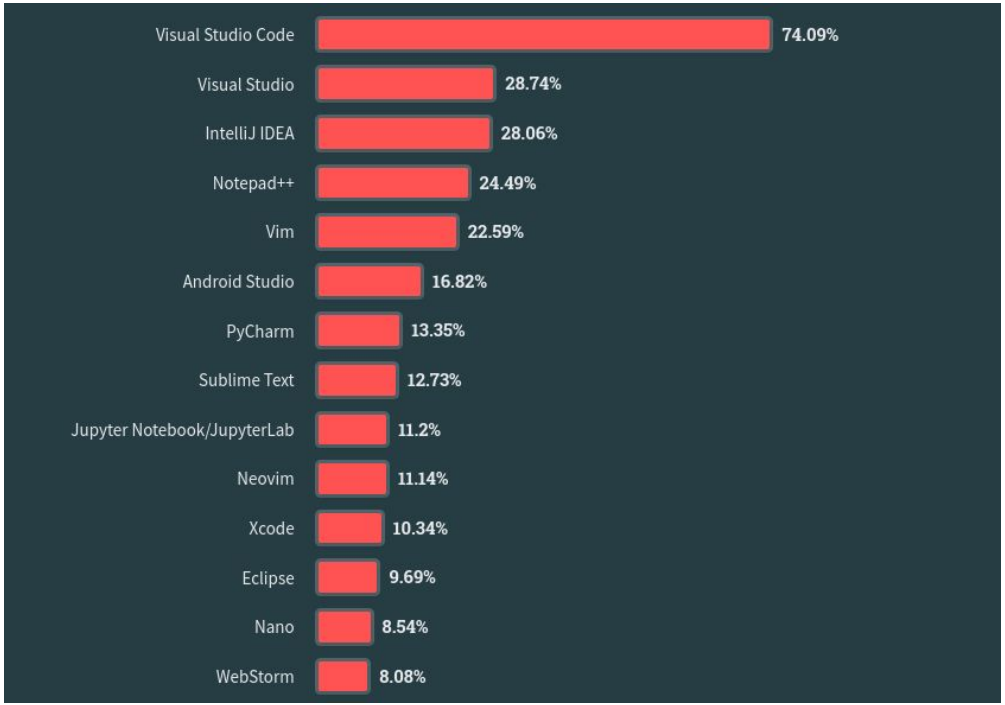
6. Debug simple code.

Stack Overflow's 2023 developer survey



Professional developers most popular programming languages

Stack Overflow's 2023 developer survey



Professional developers most popular Integrated Development Environment

Class Activity

Install VSCode and configure to run Python.

Programming basics

- Computer hardware architecture
- Interpreted vs Compiled language
- What is a Program?
- What goes wrong? (syntax, logic and semantic errors)
- Debugging vs Testing
- Your learning journey and discovery

Python basics

1. Variables, expressions and statements
2. Conditional execution
3. Functions
4. Iteration
5. Strings
6. Files
7. Lists
8. Dictionaries
9. Tuples
10. Regular expressions

1. Variables, Expressions and Statements

- Values and types: values belong to different types.
- Variable: a name that refers to a value. It's value is given via an assignment statement.
- Variable names and keywords.
- Statement: a unit of code that the Python interpreter can execute.
- Operators and operands: special symbols that represent computations like addition and multiplication. The values the operator is applied to are called operands.
- Expression: a combination of values, variables and operators.
- Order of operations: **PEMDAS**
- Modulus operator **%** and string operations (concatenation **+** vs arithmetic)
- Requesting user input.
- Comments.

Re: variable names and keywords

Python reserves 35 keywords:

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

Case Identifiers

camelCase or lowerCamelCase : e.g. in Java variable names.

PascalCase or UpperCamelCase: e.g. in Java class names.

snake_case: e.g. in Python variable and function names.

Camel_Snake_Case

kebab-case: e.g. in Racket names, and file names.

flatcase, lazycase or mumblecase: e.g. in Java package names.

UPPER_CASE or SCREAM_CASE: e.g. in C constant names.

UPPERFLATCASE

COBOL-CASE or TRAIN-CASE

_underscoreNotation

HTTP-Header-Case

Class Exercise 1

1. Write a program that asks a user for their name and then welcomes them.

2. Improve your program to prompt the user for their number of work hours in a day and amount earned per hour, to compute daily gross pay.

3. Given width = 17, and height = 14, for each of the following expressions, write the value of the expression and the type (of the value of the expression).

- 1. width//2
- 2. width/2.0
- 3. height/3
- 4. 1 + 2 * 5

2. Conditional execution

- Boolean expression: an expression that is either true or false. Operators include **!=**, **>=**, **<=**, **>**, **<**, **==**, **is**, **is not**.
- Logical operators: **and**, **or**, and **not**
- Conditional execution: **if**
- Alternative execution: **if**, **else**
- Chained conditionals: **if**, **elif**, **else**
- Nested conditionals: **if**, **else**(**if**, **else**) or **if**(**if**)
- Catching exceptions using **try** and **except**

Class Exercise 2

1. Rewrite your gross pay program using try and except so that your program handles non-numeric input gracefully by printing a message and exiting the program.

2. Write a program to prompt for a CGPA score between 0.0 and 5.0. If the score is out of range, print an error message. If the score is within range, print a grade using the 1st class, 2nd class upper and lower, 3rd class standards.

3. Functions

- Function calls
- Built-in functions e.g. max, min, len.
- Type conversion functions
- Math functions e.g. math.sin, math.pow, math.log10, math.pi, math.sqrt
- Random numbers e.g. random.random, random.randint, random.choice
- Creating a custom function
- Parameters and arguments.
- Result/Output functions vs void functions
- Body, dot notation, function definition, import statement, return value, argument, parameter, flow of execution.

Class Exercise 3

1. Rewrite your gross pay program by creating a function called **computepay** that takes two parameters **hours** and **rate**.

2. Rewrite the CGPA program from the previous exercise using a function called **computegrade** that takes a score as its parameter and returns a grade as a string.

4. Iteration

- Updating variables
- The while statement
- Infinite loops
- continue vs break
- Definite loops using for
- Counting and summing loops using counters and accumulators
- Maximum and minimum loops

Class Exercise 4

1. Find the largest value in the list of integers [5, 1, 43, 52, 12, 64, 33]

2. Create a simple version of Python built-in min() function

3. Write a program which repeatedly reads numbers until the user enters “done”. Once “done” is entered, print out the total, count, and average of the numbers. If the user enters anything other than a number, detect their mistake using try and except and print an error message and skip to the next number.

5. Strings

- A string is a sequence of characters.
- String indexes
- String traversal using while and for loops
- String slices
- Immutable characteristic of a string
- The **in** operator
- Looping and counting string variables
- String comparison
- String methods and invocation
- Parsing strings
- Format operator

Class Exercise 5

1. Write a while loop that starts at the last character of a string and works its way backwards to the first character in the string, printing each letter on a separate line, except backwards.

2. Take the following Python code that stores a string:

   str = 'X-DSPAM-Confidence:0.8475'

   Use find and string slicing to extract the portion of the string after the colon character and then use the float function to convert the extracted string into a floating point number.