**WHY IS PYTHON BEST SUITED FOR COMPETITIVE CODING?**

**Introduction**

**Competitive Coding**

Competitive Coding could also be referred to as Competitive Programming is a programming competition usually held over the internet or a local network. It can also be termed as a mind sport. Participants are presented with a set of logical or mathematical problems and are required to write computer programs capable of solving each problem. Contestants are referred to as sport programmers. Top popular programming competitions held on regular basis are: **ACM–ICPC (**International Collegiate Programming Contest**), Codejam** (organized by Google), **Hackercup** (organized by Facebook).

**Product Based Companies**

Competitive programming is recognized and supported by several multinational software and Internet companies. It also serves as a platform to hunt for coders into their companies. Example of such industries are Google, Facebook.

**PYTHON**

**History**

Python was created by Guido van Rossum in the late 1980s at Centrum Wiskunde & Informatica (CWI) in the Netherlands to take over from the ABC language. Its implementation began in December 1989. The current stable version of Python is 3.0 and was released on 3 December 2008. [40] and 2.7.x version series. Although python 3 is here, later versions of python 2 are being improved to enhance their performance under concurrent workloads.

**Why Use PYTHON?**

One important driver of competitive coding is SPEED of the programmer or coder. Python is the ultimate choice to consider in competitive coding where speed is a major factor. Coders are judged mostly based on number of problems solved and the time spent for writing successful solutions. Other factors may include quality of output produced, execution time, program size, etc.

Codes written in Python require less typing compared to other conventional programming languages like C, C++, JAVA.

Another important advantage is that python provides its users with a wide variety of functionality, packages and libraries which act like a supplement to the programmer’s mental ability.

Other feature that make Python the best choice for competitive coding are:

* Variable Independence

In python there is no need to declare variables before assigning values to them unlike java and C++ which requires that variables are declared before they are assigned.

|  |  |  |
| --- | --- | --- |
| **Python** | **Java** | **C++** |
| **msg = “Hello World”**  **print(msg)**  **Output = Hello World** | **public class () {**  **public static void main(String[]arg) {**  **string msg = “Hello World”**  **System.out.println(msg);**  **Output = Hello World**  **}**  **}** | **using namespace std;**  **int main (){**  **string msg = “Hello World”;**  **cout << msg <<endln;**  **return 0**  **}** |

The sample code given above demonstrates how the statement “Hello World” is written in Python, Java and C++. It clearly shows that python has a very simple syntax which can be easy read and understand. While java and C++ require classes to be called before writing a main code python doesn’t need all that. This saves a lot of time in coding.

* Common Functions like sorted, min, max, count etc.

The min/max function helps us to find the minimum/maximum element from a list.The Sorted function allows us to sort a list and count function helps us to count the number of occurences of a particular element in a list. The best thing is that we can be rest assured that the python libraries use the best possible algorithms for each of the above operations. For example the sorted function is a very special sorting algorithm called TIMSORT that has a worst case time complexity of **O(n log n)** which is the best a sorting algorithm can offer. Common Functions like sorted, min, max, count etc

|  |  |  |
| --- | --- | --- |
| **Python** | **Java** | **C++** |
| **arr = [10, 76, 87, 45, 22, 87, 90, 87, 66, 84, 87]**  **print("Maximum = ",max(arr))**  **print("Minimum = ",min(arr))**  **print("The sorted array is = ",sorted(arr))**  **print('Number of occurrences of 87 is = ',arr.count(87))** | **import java.util.ArrayList;**  **import java.util.Collections;**  **import java.util.List;**  **public class GFG {**  **public static Integer findMin(List<Integer> list)**  **{**  **if (list == null || list.size() == 0) {**  **return Integer.MAX\_VALUE;**  **}**  **List<Integer> sortedlist = new ArrayList<>(list);**  **Collections.sort(sortedlist);**  **return sortedlist.get(0);**  **}**    **public static Integer findMax(List<Integer> list)**  **{**  **if (list == null || list.size() == 0) {**  **return Integer.MIN\_VALUE;**  **}**    **List<Integer> sortedlist = new ArrayList<>(list);**  **Collections.sort(sortedlist);**    **return sortedlist.get(sortedlist.size() - 1);**  **}**  **public static void main(String[] args)**  **{**    **List<Integer> list = new ArrayList<>();**    **list.add(44);**  **list.add(11);**  **list.add(22);**  **list.add(33);**  **System.out.println("Min: " + findMin(list));**  **System.out.println("Max: " + findMax(list));**  **}**  **}** | **#include <iostream>**  **#include <vector>**  **#include <algorithm>**  **using namespace std;**  **void countSort(vector <int>& arr)**  **{**  **int max = \*max\_element(arr.begin(), arr.end());**  **int min = \*min\_element(arr.begin(), arr.end());**  **int range = max - min + 1;**    **vector<int> count(range), output(arr.size());**  **for(int i = 0; i < arr.size(); i++)**  **count[arr[i]-min]++;**    **for(int i = 1; i < count.size(); i++)**  **count[i] += count[i-1];**    **for(int i = arr.size()-1; i >= 0; i--)**  **{**  **output[ count[arr[i]-min] -1 ] = arr[i];**  **count[arr[i]-min]--;**  **}**    **for(int i=0; i < arr.size(); i++)**  **arr[i] = output[i];**  **}**  **void printArray(vector <int> & arr)**  **{**  **for (int i=0; i < arr.size(); i++)**  **cout << arr[i] << " ";**  **cout << "\n";**  **}**  **int main()**  **{**  **vector<int> arr = {-5, -10, 0, -3, 8, 5, -1, 10};**  **countSort (arr);**  **printArray (arr);**  **return 0;**  **}** |

* Lists in python combine the best aspects of arrays and linked lists

Python lists provide the unique functionality of deleting specific elements while keeping the memory locations in a contiguous manner. This feature renders the concept of Linked lists null and void. Moreover, Insertions can be performed at any desired locations.

|  |
| --- |
| **# Python code to demonstrate list operations**  **arr = [00, 11, 22, 33, 44, 55, 66, 77, 88, 99]**  **# deletion via index position**  **del arr[5]**  **print(arr)**  **# deletion via specifying particular element**  **arr.remove(22)**  **print(arr)**  **# insertion at any arbitrary position**  **arr[-1] = "A random number"**  **print(arr)**  **# concept of sub-lists**  **k = arr[:2]**  **print(k)** |

* Unique list operations – Backtracking, Sub-Lists.

In case we are not sure about the list size then we can use the index position of -1 to access the last element. Similarly -2 can be used for second last element and so on. Thus we can back track a list. Also we don’t have to specify any particular list size so it also works like a dynamic allocation array.

A specific portion of a list can be extracted without having to traverse the list as is seen in the above example. A very astonishing fact about lists is that it can hold different datatypes. Gone are the days where lists used to be a homogeneous collection of data elements!!

* Functions can return more than one value

Typically functions in other programming languages can return only one value but in python we can return more than one value!! as is seen in the following code snippet.

|  |
| --- |
| **# Python code to demonstrate that a function**  **# can easily return multiple values.**  **def multi\_return(\*arr):**  **k1 = arr[0]**  **k2 = arr[1]**  **return k1,k2**    **a,b = multi\_return(11,22)**  **print(a,' ',b)**  **a,b = multi\_return(55,66,77,88,99)**  **print(a,' ',b)** |

* Flexible number of arguments to a function

Arguments to a function may be passed in the form of a list whose size may vary every time we need to call the function. In the above example we first called the function with 2 arguments and then with 5 arguments!!

* If, else and for loops are much more User Friendly.

The if-else statement of python allows us to search for a particular element in a list without the need of traversing the entire list and checking each element.

Some programming languages have a concept of a foreach loop which is slightly different from a for loop. It allows us to traverse a list where the loop variable takes upon the list values one by one. Python incorporates foreach loop concept in the for loop itself.

|  |
| --- |
| **# Python code to demonstrate quick searching**  **arr = [1, 2, 3, 4, 5, 6, 7, 8, 9]**  **# searching made easy**  **if 3 in arr:**  **print("YES")**  **else:**  **print("NO")**  **#foreach loop**  **for i in arr:**  **print(i,end = ' ')** |

* Code Indentation

Python blocks of code are distinguished on the basis of their indentation. This provides better code readability and instills in us a good habit of indenting our code.

* Concept of Sets and Dictionaries.

A Set is an unordered collection data type that is iterable, mutable, and has no duplicate elements. Its like a list that doesn’t allow duplicate elements.

A dictionary is like a list whose values can be accessed by user defined keys instead of conventional numeric index values

|  |
| --- |
| **# Python code to demonstrate use of dictionaries**  **# and sets.**  **a = {'a','b','c','d','e','a'}**  **# the second 'a' is dropped to avoid repetition**  **print(a)**  **dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}**  **print("dict['Name']: ", dict['Name'])**  **print("dict['Age']: ", dict['Age'])** |

* Robust input statements.

In competitive coding we are often required to take ‘n’ space separated integers as input and preferably save them in a list/array. Python provides functionality to do it all in a single line of code.!!

|  |
| --- |
| **# Python code to demonstrate how to take space**  **# separated inputs.**  **arr = [int(a) for a in input().strip().split(' ')]**  **print(arr)** |