Тест 1

When we talk about programming languages the first thing that comes to our mind is languages like C,

C++, Java, Python etc. But those languages hide the actual working i.e., that abstracts many things from

users. But there is a language that really lies on basic concepts behind the programming or interaction

between computer hardware.

What is Assembly Language.

Assembly language is a low-level language that helps to communicate directly with computer hardware.

It uses mnemonics to represent the operations that a processor has to do. Which is an intermediate

language between high-level languages like C++ and the binary language. It uses hexadecimal and binary

values, and it is readable by humans.

How Assembly Language Works.

Assembly languages contain mnemonic codes that specify what the processor should do. The mnemonic

code that was written by the programmer was converted into machine language (binary language) for

execution. An assembler is used to convert assembly code into machine language. That machine code is

stored in an executable file for the sake of execution.

It enables the programmer to communicate directly with the hardware such as registers, memory

locations, input/output devices or any other hardware components. Which could help the programmer

to directly control hardware components and to manage the resources in an efficient manner.

How to execute Assembly Language.

Write assembly code: Open any text editor in device and write the mnemonic codes in it and save the

file with a proper extension according to your assembler. Extension can be .asm, .s, .asmx.

Assembling the code: Convert your code to machine language using an assembler.

Generating object file: It will generate an object file corresponding to your code. It will have an

extension .obj.

Linking and creating executables: Our assembly language might contain multiple source codes. And we

have to link them to libraries to make it executable. We can use a linker like lk for this purpose.

Running program: After creating an executable file we can run it as usual. It will depend on the software

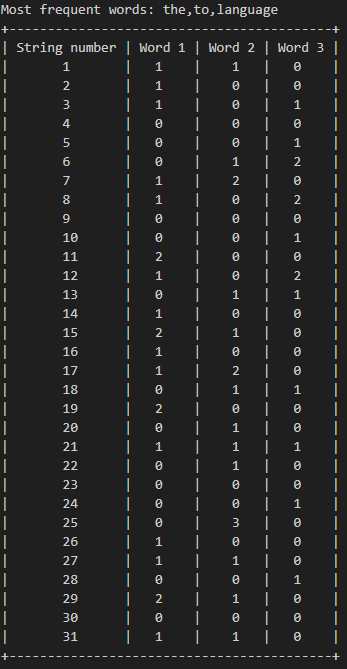
that how to run the program.

Components of Assembly Language.

Registers: Registers are the fast memory locations situated inside the processor. Which helps ALU to

perform arithmetic operations and temporary storing of data. Example: Ax (Accumulator), Bx, Cx.

Command: An instruction in assembly code known as a command informs the assembler what to do.



|  |  |  |  |
| --- | --- | --- | --- |
| String number | The | To | Language |
| 1 | 1 | 1 | 0 |
| 2 | 1 | 0 | 0 |
| 3 | 1 | 0 | 1 |
| 4 | 0 | 0 | 0 |
| 5 | 0 | 0 | 1 |
| 6 | 0 | 1 | 2 |
| 7 | 1 | 2 | 0 |
| 8 | 1 | 0 | 2 |
| 9 | 0 | 0 | 0 |
| 10 | 0 | 0 | 1 |
| 11 | 2 | 0 | 0 |
| 12 | 1 | 0 | 2 |
| 13 | 0 | 1 | 1 |
| 14 | 1 | 0 | 0 |
| 15 | 2 | 1 | 0 |
| 16 | 1 | 0 | 0 |
| 17 | 1 | 2 | 0 |
| 18 | 0 | 1 | 1 |
| 19 | 2 | 0 | 0 |
| 20 | 0 | 1 | 0 |
| 21 | 1 | 1 | 1 |
| 22 | 0 | 1 | 0 |
| 23 | 0 | 0 | 0 |
| 24 | 0 | 0 | 1 |
| 25 | 0 | 3 | 0 |
| 26 | 1 | 0 | 0 |
| 27 | 1 | 1 | 0 |
| 28 | 0 | 0 | 1 |
| 29 | 2 | 1 | 0 |
| 30 | 0 | 0 | 0 |
| 31 | 1 | 1 | 0 |

Табл. 1 Ожидаемый результат Рис. 1 Результат работы программы

Тест 2

C++ is a general-purpose programming language and is widely used nowadays for competitive

programming. It has imperative, object-oriented, and generic programming features. C++ runs on lots of

platforms like Windows, Linux, Unix, Mac, etc.

Before explaining the basics of C++, we would like to clarify two more ideas: low-level and high-level. To

make it easy to understand, let’s consider this scenario – when we go to the Google search engine and

search for some queries, Google displays some websites according to our question. Google does this for us

at a very high level. We don’t know what’s happening at the low level until we look into Google servers (at

a low level) and further to the level where the data is in the form of 0s/1s. The point we want to make here

is that a low level means nearest to the hardware, and a high level means farther from the hardware with a

lot of layers of abstraction. C++ is considered a low-level language as it is closer to hardware than most

general-purpose programming languages. However to become proficient in any programming language,

one Firstly needs to understand the basics of that language.

Basics of C++ Programming

Basic Syntax and First Program in C++

Learning C++ programming can be simplified into writing your program in a text editor and saving it with

the correct extension(.CPP, C, CP), and compiling your program using a compiler or online IDE. The “Hello

World” program is the first step toward learning any programming language and is also one of the simplest

programs you will learn.

We can learn more about C++ Basic Syntax here – C++ Basic Syntax

Basic Input and Output in C++

C++ comes with libraries that provide us with many ways for performing input and output. In C++ input and

output are performed in the form of a sequence of bytes or more commonly known as streams. The two

methods cin and cout are used very often for taking inputs and printing outputs respectively. These two are

the most basic methods of taking input and output in C++.

To learn more about basic input and output in C++, refer to this article – Basic Input/Output in C++

Comments in C++

A well-documented program is a good practice for a programmer. It makes a program more readable and

error finding becomes easier. One important part of good documentation is Comments. In computer

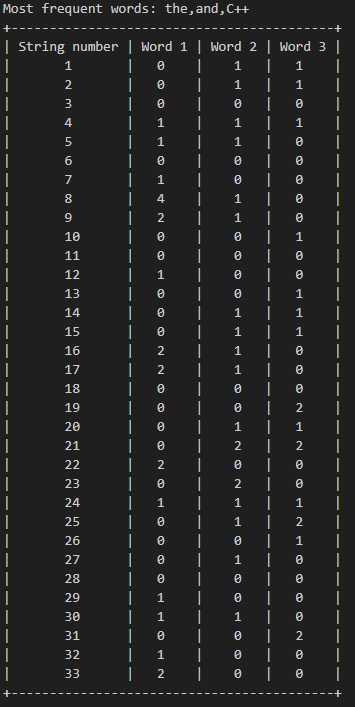
programming, a comment is a programmer-readable explanation or annotation in the source code of a

computer program. These are statements that are not executed by the compiler and interpreter.

We can learn more about C++ comments in this article – C++ Comments

All variables use data type during declaration to restrict the type of data to be stored. Therefore, we can

say that data types are used to tell the variables the type of data they can store.



|  |  |  |  |
| --- | --- | --- | --- |
| String number | The | And | C++ |
| 1 | 0 | 1 | 1 |
| 2 | 0 | 1 | 1 |
| 3 | 0 | 0 | 0 |
| 4 | 1 | 1 | 1 |
| 5 | 1 | 1 | 0 |
| 6 | 0 | 0 | 0 |
| 7 | 1 | 0 | 0 |
| 8 | 4 | 1 | 0 |
| 9 | 2 | 1 | 0 |
| 10 | 0 | 0 | 1 |
| 11 | 0 | 0 | 0 |
| 12 | 1 | 0 | 0 |
| 13 | 0 | 0 | 1 |
| 14 | 0 | 1 | 1 |
| 15 | 0 | 1 | 1 |
| 16 | 2 | 1 | 0 |
| 17 | 2 | 1 | 0 |
| 18 | 0 | 0 | 0 |
| 19 | 0 | 0 | 2 |
| 20 | 0 | 1 | 1 |
| 21 | 0 | 2 | 2 |
| 22 | 2 | 0 | 0 |
| 23 | 0 | 2 | 0 |
| 24 | 1 | 1 | 1 |
| 25 | 0 | 1 | 2 |
| 26 | 0 | 0 | 1 |
| 27 | 0 | 1 | 0 |
| 28 | 0 | 0 | 0 |
| 29 | 1 | 0 | 0 |
| 30 | 1 | 1 | 0 |
| 31 | 0 | 0 | 2 |
| 32 | 1 | 0 | 0 |
| 33 | 2 | 0 | 0 |

Табл. 2 Ожидаемый результат Рис.2 Результат работы программы

Тест 3

This Programming Language Python Tutorial is very well suited for beginners and also for experienced

programmers. This specially designed free Python tutorial will help you learn Python programming most

efficiently, with all topics from basics to advanced (like Web-scraping, Django, Learning, etc.) with

examples.

What is Python?

Python is a high-level, general-purpose, and very popular programming language. Python programming

language (latest Python 3) is being used in web development, and Machine Learning applications, along

with all cutting-edge technology in Software Industry. Python language is being used by almost all

tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc.

Writing your first Python Program to Learn Python Programming

There are two ways you can execute your Python program:

First, we write a program in a file and run it one time.

Second, run a code line by line.

Here we provided the latest Python 3 version compiler where you can edit and compile your written code

directly with just one click of the RUN Button. So test yourself with Python first exercises.

Welcome to the Python tutorial section! Here, we’ll cover the essential elements you need to kickstart your

journey in Python programming. From syntax and keywords to comments, variables, and indentation, we’ll

explore the foundational concepts that underpin Python development.

In this segment, we delve into the fundamental aspects of handling input and output operations in Python,

crucial for interacting with users and processing data effectively. From mastering the versatile print()

function to exploring advanced formatting techniques and efficient methods for receiving user input, this

section equips you with the necessary skills to harness Python’s power in handling data streams seamlessly.

Python offers, enabling you to manipulate and manage data with precision and flexibility. Additionally, we’ll

delve into the dynamic world of data conversion with casting, and then move on to explore the versatile

collections Python provides, including lists, tuples, sets, dictionaries, and arrays.

By the end of this section, you’ll not only grasp the essence of Python’s data types but also wield them

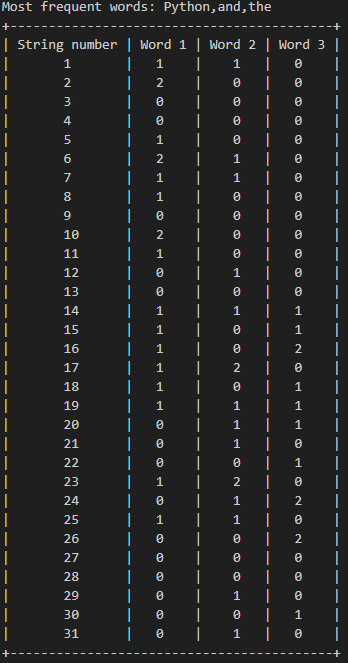
proficiently to tackle a wide array of programming challenges with confidence.

From performing basic arithmetic operations to evaluating complex logical expressions, we’ll cover it all.

We’ll delve into comparison operators for making decisions based on conditions, and then explore bitwise

operators for low-level manipulation of binary data. Additionally, we’ll unravel the intricacies of assignment

operators for efficient variable assignment and updating.



|  |  |  |  |
| --- | --- | --- | --- |
| String number | Python | And | The |
| 1 | 1 | 1 | 0 |
| 2 | 2 | 0 | 0 |
| 3 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 |
| 5 | 1 | 0 | 0 |
| 6 | 2 | 1 | 0 |
| 7 | 1 | 1 | 0 |
| 8 | 1 | 0 | 0 |
| 9 | 0 | 0 | 0 |
| 10 | 2 | 0 | 0 |
| 11 | 1 | 0 | 0 |
| 12 | 0 | 1 | 0 |
| 13 | 0 | 0 | 0 |
| 14 | 1 | 1 | 1 |
| 15 | 1 | 0 | 1 |
| 16 | 1 | 0 | 2 |
| 17 | 1 | 2 | 0 |
| 18 | 1 | 0 | 1 |
| 19 | 1 | 1 | 1 |
| 20 | 0 | 1 | 1 |
| 21 | 0 | 1 | 0 |
| 22 | 0 | 0 | 1 |
| 23 | 1 | 2 | 0 |
| 24 | 0 | 1 | 2 |
| 25 | 1 | 1 | 0 |
| 26 | 0 | 0 | 2 |
| 27 | 0 | 0 | 0 |
| 28 | 0 | 0 | 0 |
| 29 | 0 | 1 | 0 |
| 30 | 0 | 0 | 1 |
| 31 | 0 | 1 | 0 |

Табл. 3 Ожидаемый результат Рис. 3 Результат работы программы

Результат тестов и выводы:

Для проверки правильности работы программы были подобраны 3 текста, посчитаны три наиболее часто встречающихся слова в каждом тексте и их количество в каждой строке соответствующих текстов, после чего были составлены три таблицы с ожидаемым результатом работы программы.

Рядом с таблицами были прикреплены снимки результата работы программы. Сравнив таблицы с соответствующими им снимками, видим, что ожидаемый результат совпадает с результатом работы программы для всех трех тестов, из чего можно сделать вывод о корректности работы программы.