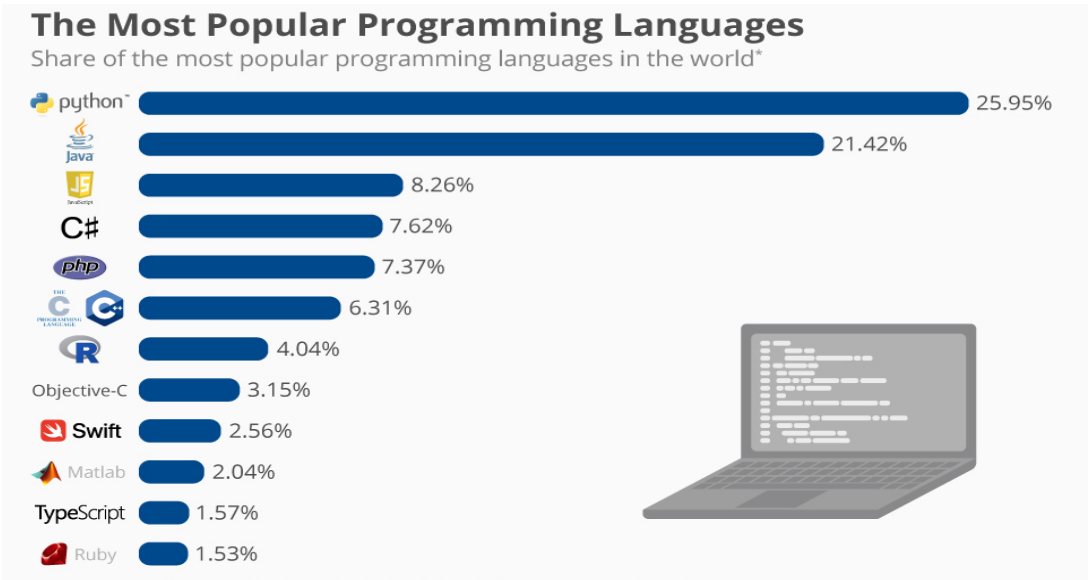
NAME – INDRANIL BAIN

SUBJECT – SOFTWARE ENGINEERING LAB

GROUP – GX (No. 6)
ROLL NO. – 2020CSB039

Coding in Software Engineering :

Coding is the process of writing the source code for a software application. In software engineering, coding is typically just one phase of the software development process, which also includes activities such as requirement gathering, design, testing, and maintenance. When writing code, software engineers follow a set of coding standards and guidelines to ensure that the code is easy to read, maintain, and debug. They may also use software development tools such as text editors, integrated development environments (IDEs), and version control systems to help with the coding process. In addition to writing code, software engineers may also be responsible for performing code reviews to ensure that the code meets the required standards and is free of defects. They may also write documentation to explain the operation and purpose of the code, and to provide guidance for other developers who may need to modify or maintain the code in the future. Overall, the goal of coding in software engineering is to produce high-quality, reliable, and maintainable software that meets the needs of the users.



Coding's objectives:

To convert a system's design into computer language format:
The process of converting a system's design into a computer language that can be executed by a computer and that can carry out duties as described by the design of operation during the design phase is known as coding.

To lower the price of subsequent phases:
Effective code can considerably lower the cost of testing and maintenance.

Making the programme easier to read:
The programme must to be simple to read and comprehend. It improves code comprehension, and having readability and understandability as a specific goal of the coding activity itself can assist in creating software that is easier to maintain.

Characteristics of Programming Language:

Readability: A suitable high-level language should enable the creation of programmes in a manner that resembles a simple English explanation of the underlying operations. It is possible to code in a largely self-documenting manner. Being essentially machine-independent, high-level languages should make it simple to create portable software.

Generality: Most high-level languages enable the creation of a large number of programmes, saving the programmer from having to become fluent in numerous different languages.

Shortness: Languages should be able to use less code to implement algorithms. Programs written in high-level languages are frequently much shorter than those written in low-level languages.

Error-checking: When creating a computer programme, a programmer is likely to make a lot of mistakes.

Coding Standards:

Indentation: Proper and consistent indentation is crucial for creating programmes that are simple to read and maintain.

Use of indentation is advised for:

1. In a control structure, such a loop or a select statement, emphasise the body.
2. Put more emphasis on a conditional statement's body
3. Incorporate a new scope block.

Inline remarks: Inline comments that examine how a subroutine works or important elements of an algorithm should be utilised regularly.

Limitations on the use of global: These regulations specify which types of data can and cannot be deemed global.

Structured Programming: Modular or structured programming techniques must be employed. Except as specified in the FORTRAN Standards, "GOTO" commands should not be used since they produce "spaghetti" code, which is difficult to comprehend and maintain.

Coding Guidelines :

1. **Line Length:** Keeping source code lines at or below 80 characters is thought to be a recommended practise. On some terminals and tools, lines that are longer than this may not be clearly seen. If a line is longer than 80 columns, certain printers will truncate it.

2. **Spacing:** A line of code's readability can be increased by using spaces in the right places.

3. **The code must be thoroughly documented:** As a general guideline, for every three source lines, there ought to be at least one remark line.

Bad:

cost=price+(price*sales_tax) fprintf(stdout "The total cost is %5.2fln", cost);

Better: cost = price + (price * sales_tax) fprintf (stdout, "The total cost is %5.2f]n",cost);

4. **No function should be longer than 10 source lines:** Due to the possibility that it performs numerous different activities, a really long function is typically exceedingly challenging to understand. The same logic also applies to long functions.

-:-

Part 2 – Case study

In the coding part of software engineering a case study refers that how to choose the suitable language for the environment of the type of developing we are doing.

Like in Web development we needt to make the design and the prototype of the website then we need to choose the language of the website like we can use HTML,CSS and we can use Nodejs over PHP for better development of the website and on the other hand for software development we can use new powerfull languages for developing like we can use Go Lang instead of c++ and C.

In this case study we can say that main point of the case study is to select the language of the coding environment, write good quality code and documenting the code properly.

In a case study of the google meet I have found that if the software was written in Go lang it could be more reliable and less crashed in running time. The software was wriiten in c++ that's why it is not that strong to be specified.