1. LOC: A line of code (LOC) is any line of text in a code that is not a comment or blank line, and also header lines, in any case of the number of statements or fragments of statements on the line. LOC consists of all lines containing the declaration of any variable, and executable and non-executable statements.

LOC = Function Points\*Conversion Factor

1. **Features** of Lines of Code (LOC)

* Change Tracking: Variations in LOC as time passes can be tracked to analyze the growth or reduction of a codebase, providing insights into project progress.
* Limited Representation of Complexity: Despite LOC provides a general idea of code size, it does not accurately depict code complexity. It is possible for two programs having the same LOC to be incredibly complex.
* Ease of Computation: LOC is an easy measure to obtain because it is easy to calculate and takes little time.
* Easy to Understand: The idea of expressing code size in terms of lines is one that stakeholders, even those who are not technically inclined, can easily understand.

1. **Advantages** of Lines of Code (LOC)

* Effort Estimation: LOC is occasionally used to estimate development efforts and project deadlines at a high level. Although caution is necessary, project planning can begin with this.
* Comparative Analysis: High-level productivity comparisons between several projects or development teams can be made using LOC. It might provide an approximate figure of the volume of code generated over a specific time frame.
* Benchmarking Tool: When comparing various iterations of the same program, LOC can be used as a benchmarking tool. It may bring information on how modifications affect the codebase’s total size.

1. **Disadvantages** of Lines of Code (LOC)

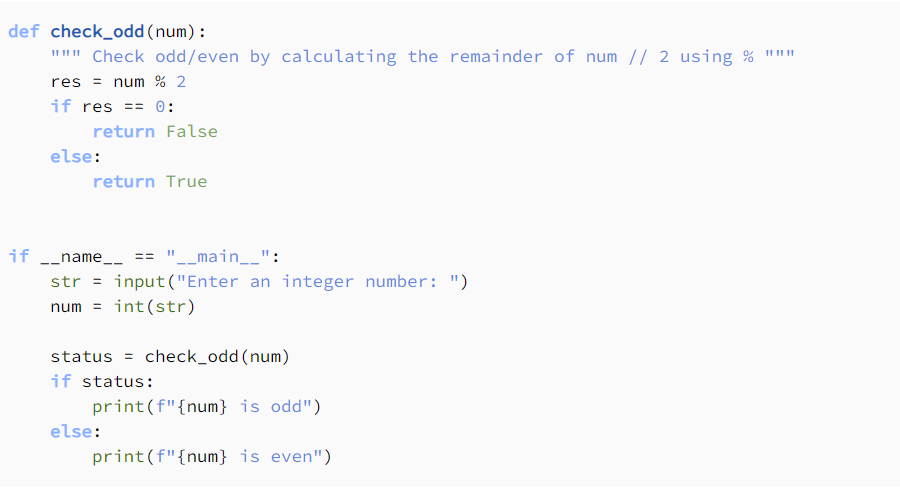
* Challenges in Agile Work Environments: Focusing on initial LOC estimates may not adequately reflect the iterative and dynamic nature of development in agile development, as requirements may change.
* Not Considering Into Account External Libraries: Code from other libraries or frameworks, which can greatly enhance a project’s overall usefulness, is not taken into account by LOC.
* Challenges with Maintenance: Higher LOC codebases are larger codebases that typically demand more maintenance work.

1. **Calculation** method

* There are two subcategories of LOC:
  + Physical LOC
  + Logical LOC
* Physical LOC counts the number of actual lines of code separated by an end marker. For example, statements between semicolons constitute a line in C programs.
* In contrast, logical LOC examines a single physical line of code and counts standalone statements in it. For instance, x=int(input(“Enter your age: “)) is one physical line but two logical since it has two statements:
  + taking user input via input
  + typecasting via int

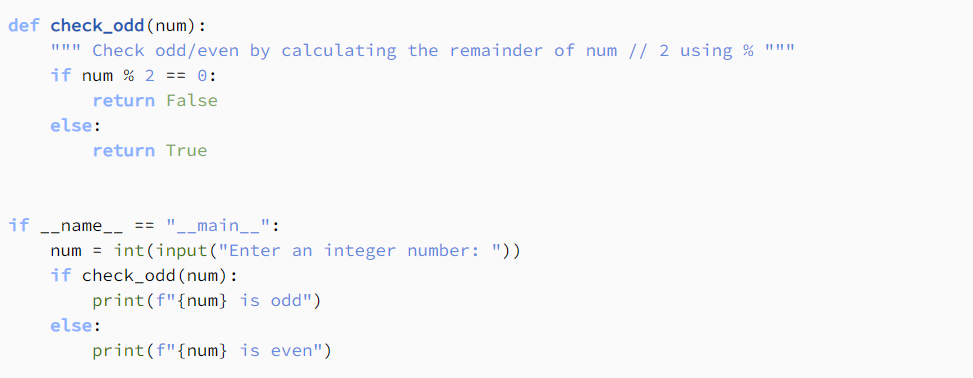
1. **Example**

* Let’s consider the following Python code snippet for checking if an integer is odd or even:



In this code, physical and logical LOCs are 14.

Now, let’s consider another version of the same code. In it, we combined the first two lines of \_\_main\_\_, as well as the third and fourth lines. We also made check\_odd more concise by removing res and replacing it with num % 2 in the condition of the if statement:

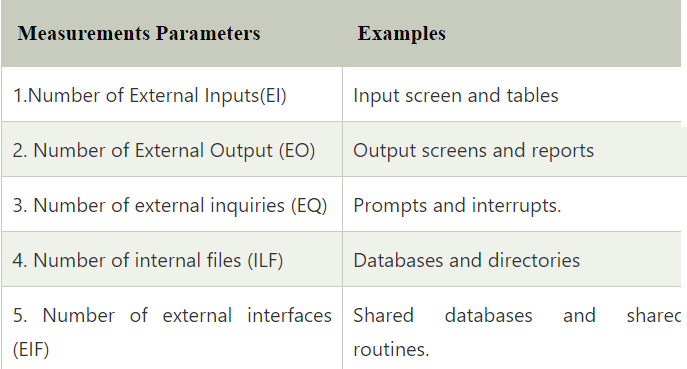


In this code, the physical LOC is 11, but the logical LOC remains 14. This is because the collapsed lines in \_\_main\_\_ consist of two logically executable statements. The first entails taking input and casting a string to an integer, while the second one stores the result of check\_odd and compares it to True. Similar goes for if num % 2 == 0 in check\_odd.

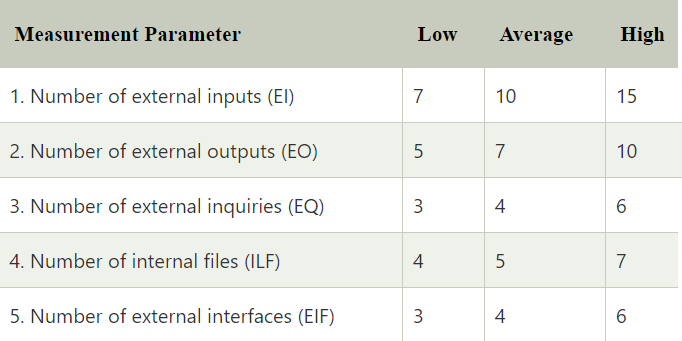
1. FP: **Definition**: Function Point is a method of measuring the size of software based on the functionality and value it provides, rather than the number of lines of code. This makes FP independent of the programming language and technology used.
2. **Usage**: FP is used to:

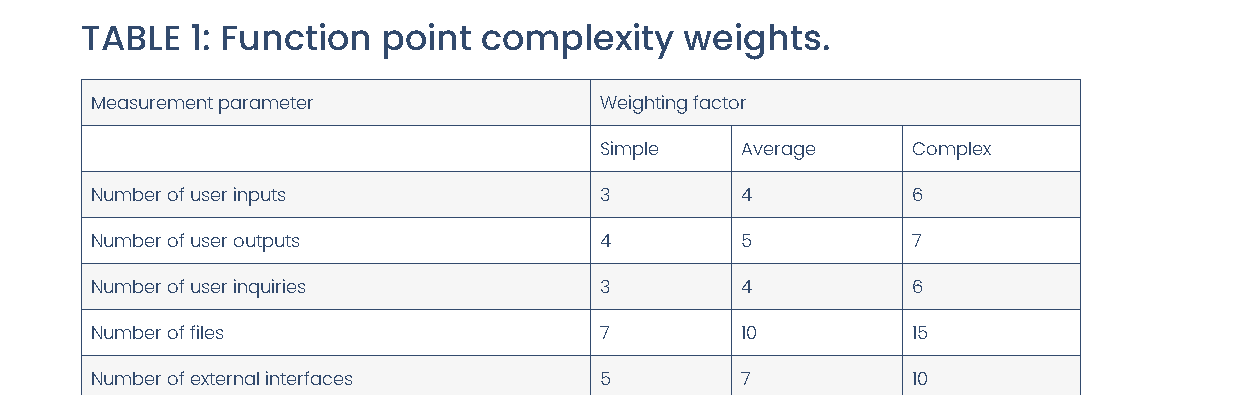
* Estimate the effort required for software development and maintenance.
* Compare productivity and efficiency between projects or development teams.
* Manage projects and monitor the progress of software development.

1. **Calculation**: FP is calculated based on five main types of functions:



Each type of function is assessed based on complexity (low, medium, high) and assigned a corresponding weight. The total number of FPs is calculated by multiplying the total number of unadjusted functions (UFP) by an adjustment factor (CAF) based on factors that affect the project





F.P = UFP x VAF

UFP : Unadjusted Fucntional Point

VAF : Value Adjustment Factor

Weighting: EI, EO, EQ, ILF, EIF

UFP = Weighting\*N(EI) + Weighting\*N(EO) + Weighting\*N(EQ) + Weighting\*N(ILF) + Weighting\*N(EIF)

VAF = 0.65 + (0.01\* ∑Fi)

1. **Advantages**:

* Independent of the programming language.
* Reflects the functionality and value that the software provides.
* Useful for project management and resource estimation.

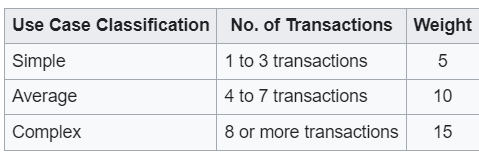
1. **Disadvantages**:

* Requires experience and specialized knowledge for accurate assessment.
* It may take time to collect and analyze the necessary data.
* Does not fully reflect non-functional factors such as performance or maintainability.

1. UCP: Use case points (UCP or UCPs) is a software estimation technique used to forecast the software size for software development projects. UCP is used when the Unified Modeling Language (UML) and Rational Unified Process (RUP) methodologies are being used for the software design and development. The concept of UCP is based on the requirements for the system being written using use cases, which is part of the UML set of modeling techniques. The software size (UCP) is calculated based on elements of the system use cases with factoring to account for technical and environmental considerations. The UCP for a project can then be used to calculate the estimated effort for a project.
2. **Method**: The method for determining the size estimate to develop a system is based on a calculation with the following elements:

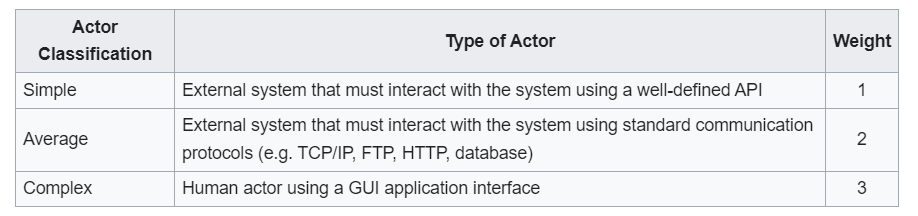
* Unadjusted Use Case Weight (UUCW) – the point size of the software that accounts for the number and complexity of use cases.
* Unadjusted Actor Weight (UAW) – the point size of the software that accounts for the number and complexity of actors.
* Technical Complexity Factor (TCF) – factor that is used to adjust the size based on technical considerations.
* Environmental Complexity Factor (ECF) – factor that is used to adjust the size based on environmental considerations.

1. **Unadjusted Use Case Weight (UUCW)**



UUCW = (Total No. of Simple Use Cases x 5) + (Total No. Average Use Cases x 10) + (Total No. Complex Use Cases x 15)

1. **Unadjusted Actor Weight (UAW)**



UAW = (Total No. of Simple actors x 1) + (Total No. Average actors x 2) + (Total No. Complex actors x 3)

1. **Use Case Points (UCP)**

**UCP = (UUCW + UAW)**

1. EVM:
2. **Definition**:

* EVM is a method that allows project managers to track the performance of their projects against project baselines, providing a way to measure progress and performance in monetary terms.

1. **Usage**: EVM is used to

* Determine if a project is on track in terms of budget and schedule.
* Forecast potential project issues.
* Make informed decisions to bring projects back on track if necessary.

1. **Calculation**: The calculation of EVM involves three key data points:

* Planned Value (PV): The budgeted cost of work scheduled.
  + Planned Value (PV): This is the estimated cost of the work that should have been completed by a certain date. It’s also known as the Budgeted Cost of Work Scheduled (BCWS). To calculate PV, you multiply the planned percentage of completed work by the project’s total budget.
  + For example, if your project budget is $100,000 and you have planned to complete 25% of the work by a certain date, the PV would be: PV=25%×$100,000=$25,000
* Actual Cost (AC): The actual cost incurred for the work performed.
  + Actual Cost (AC): This is the actual cost incurred for the work performed up to a certain date. It’s also known as the Actual Cost of Work Performed (ACWP). AC is simply the sum of all costs incurred for the work done so far.
  + For instance, if you spent $20,000 on labor and $10,000 on materials, the AC would be: AC=$20,000+$10,000=$30,000
* Earned Value (EV): The budgeted cost of work actually performed.
  + Earned Value (EV): This is the budgeted amount for the work actually completed to date. It’s also known as the Budgeted Cost of Work Performed (BCWP). To calculate EV, you determine the percentage of the total work actually completed and multiply it by the project’s total budget.
  + For example, if you have actually completed 20% of the total project work, the EV would be: EV=20%×$100,000=$20,000

1. **Advantages**:

* Provides an objective measure of project performance.
* Helps in early detection of project issues.
* Facilitates better project control and management decisions.

1. **Disadvantages**:

* Requires accurate baseline and progress data for effective use.
* Can be complex to implement and understand for those new to the technique.
* May not fully account for qualitative aspects of project performance.

1. DRE:
2. **Definition**: DRE is calculated by dividing the number of defects identified and resolved during a specific time period by the total number of defects introduced into the system. It is usually expressed as a percentage.
3. **Usage**:

* Assess the quality of the testing process.
* Identify areas of the development process that may need improvement.
* Provide insights into the effectiveness of quality assurance activities.

1. Calculation: The formula for calculating DRE is:

DRE = (Total defects found during testing/ (Total defects found during testing + Total defects found after release)) \* 100%

1. **Advantages**:

* Objective Measurement: Offers an objective way to measure the effectiveness of the defect removal process.
* Quality Improvement: Helps in identifying the effectiveness of quality assurance and testing efforts, leading to continuous improvement.

1. **Disadvantages**:

* Data Accuracy: Requires accurate data on defects, which can be challenging to obtain.
* Timeframe Limitation: May not account for defects found long after the release.
* Does Not Measure Severity: Does not differentiate between the severity of defects.

Hello teacher and everyone, the following is the presentation of group 5 on topic 4 Software Metrics Dashboard Development. First, come to the introduction of the group members. My group includes..... Next comes the presentation content including.......

Coming to the Introduction, I will introduce ........