B.Tech (CSE) 6 Semester

Software Engineering (CS1606)

Code Metrics Generator Tool

Group No. 14
CSE A

1)Keywords: Code metrics,Software Complexity,Python

2) Topic:

Code metrics is a set of software measures that provide developers better insight into the code they are developing.Metrics generator tool can be used by developers to generate code metrics data for a python code, that measures the complexity and maintainability of their managed code.
By taking advantage of code metrics, developers can understand which types and/or methods should be reworked or more thoroughly tested for a given code.

3) Introduction:

Though there are several metrics generation tools available, these tools mostly focus on languages such as C,C++,Java.Python is one of the major emerging languages.Not many tools are available for code metrics evaluation of a Python code.Thus our project focusses on building a tool for Python language code analysis.The project is inspired by the Python tool named Radon.

4) Problem Statement:

In software programming, as the design of software is realized, the number of elements and their interconnections gradually emerge to be huge, which becomes too difficult to understand at once. Software design complexity is difficult to assess without using complexity metrics and measures.The increased complexity of modern software applications also increases the difficulty of making the code reliable and maintainable.
Thus in response to above mentioned problem, the tool generates different code metrics as per the user requirements.It aims at helping the developers to not only reduce the complexity of their codes but also in building a maintainable software.
Some of the code metrics that the tool will generate are as follows:

1. <u>Cyclomatic Complexity:</u>

Cyclomatic Complexity corresponds to the number of decisions a block of code contains plus 1. This number (also called McCabe number) is equal to the number of linearly independent paths through the code. This number can be used as a guide when testing conditional logic in blocks.
The tool will analyze the AST tree of a Python program to compute Cyclomatic Complexity.

2.<u>Maintainability Index:</u>

Maintainability Index is a software metric which measures how maintainable (easy to support and change) the source code is. The maintainability index is calculated as a factored formula consisting of SLOC (Source Lines Of Code), Cyclomatic Complexity and Halstead volume.

3.Raw Metrics:

The following are the definitions employed:

1. **LOC**: The total number of lines of code. It does not necessarily correspond to the number of lines in the file.
2. **LLOC**: The number of logical lines of code. Every logical line of code contains exactly one statement.
3. **SLOC**: The number of source lines of code - not necessarily corresponding to the **LLOC**.
4. Comments: The number of comment lines. Multi-line strings are not counted as comment since, to the Python interpreter, they are just strings.
5. Multi: The number of lines which represent multi-line strings.
6. Blanks: The number of blank lines (or whitespace-only ones).

The equation **SLOC + Multi + Single comments + Blank = LOC** should always hold.

4.Halstead Metrics:

Halstead's goal was to identify measurable properties of software, and the relations between them. These numbers are statically computed from the source code:

1. $n1$ = the number of distinct operators
2. $n2$ = the number of distinct operands
3. $N1$ = the total number of operators
4. $N2$ = the total number of operands

From these numbers several measures can be calculated:

1. Program vocabulary: $n=n1+n2$
2. Program length: $N=N1+N2$
3. Calculated program length: $N'=n1log2n1+n2log2n2$
4. Volume: $V=Nlog2n$
5. Difficulty: $D=n1/2\cdot N2/n2$
6. Effort: $E=D\cdot V$
7. Time required to program: $T=E/18$ seconds
8. Number of delivered bugs: $B=V/3000$

5) Modules:

Module	Estimated LoC	Remarks
Initial user interface:	1.5k LoC	This module will have the GUI for the start up page. The initial starting page will provide the first impression of the software.
Metric selection:	0.5k LoC	The user must choose the metric to be evaluated.The above mentioned metrics will be available to the user as an option.

Module	Estimated LoC	Remarks
Input data module:	0.5k LoC	Once the metric is chosen the user must input the python program path through a user interface.
Report generation module:	1k LoC	Each metric chosen will have a separate module for calculation of metric value.Thus accordingly a metric report will be generated.
Report saving module:	1k LoC	The generated report will be saved for future reference by the user.

6) Estimated LoC:

Total estimated LoC= 4.5 k LoC

7) Specifications:

Software requirements:

The end user must follow the below requirements such as to enjoy the features provided by the system. All these functionalities need to be necessarily incorporated into the system as a part of the contract. To get the following desired output, and maintain it's following features like:

1. Portability
2. Reliability
3. Resusability
4. Flexibility
5. Performance

The software requirements to deploy the application are as follows:

1. **Environment Required:** Python 3.x environment is required, such that to be able to run the following application. This environment can be created with virtual environments like Anaconda, Miniconda, etc or Python 3.x compiler is required.
2. **Dependencies:** In-built Python and several other libraries are also required to be pre-installed before running the application. The libraries such as:
    1. Django

2. Json
3. flake8_polyfill
4. ast
5. collections
6. math
7. tokenize
8. operator
9. pytest

   Most of the libraries are pre-built-in and so, won't be required to install them manually.

Input requirements: The input files must be of Python Executable format i.e. .py . The files shouldn't be harmful or malicious files, else, it could affect the security of the system.

Hardware Requirements:

1. **Operating Systems:** The following application is platform independent and hence can run on any platform having the environment to run the application.
   Suggested Operating Sytem Distributions are:

   1. Windows 7/8/8.1/10
   2. Linux Distributions like: Debian, Ubuntu, Arch etc.

2. **Processor:** Intel Pentium, Atom, Core i3 or higher versions would be recommended.

1. **RAM:** 4 GB is sufficient for proper functionality of the software. This amount of RAM will allow the user to run the software without any lag. It would be beneficial for the user to have RAM greater than that specified in the requirements for better overall experience and enhanced multi-tasking.
2. **Graphics Card:** Use of an integrated Intel HD Graphics GPU embedded within at least 5th generation may be enough in this case.
3. **Storage:** Storage space of atleast 10 GB is required for the minimum functionality of the application along with the other already installed softwares.

8) Testing:

1. **Unit Testing / White box testing : Performed along the project**

Test Objective	Steps	Test Data	Expected Result
Successful login of the user	Enter username and password as input	A valid username and password	'Success' if the user is registered else shows 'Invalid user'
Checking if metric selection option works	Run the metric selection module separately	Selected option by the user	The selected option gets registered in the system
Checking if the input data is working	Run the input data module to check it's functioning	Data provided by the user	The input data from the user gets stored in the system
Checking if the metric calculation works	Run the metric calculation module	Input data from the system	The code metrics are generated for user data
Checking if report generation and saving works	Run the report saving module	Generated results from the system	The report is generated and saved likewise as per the user requirements

2. **System Testing/ Black box testing : Performed after the project has been completed**

Test Objective	Steps	Test Data	Expected Result
Successful login of the user	Enter username and password in login panel	A valid username and password	The user is logged in successfully
Selecting metric option	Enter the type of code metric to be evaluated	A valid choice from the available options	The option selected gets registered into the system
Providing input data	Enter the code data for which metrics are to be calculated	A syntactically correct python code	The desired metric report is generation
Report Saving	The user chooses if report to be saved for future use	Generated metric report	The report gets saved for the registered user

9) Role of each member:

Name	Task	Signature
Juhi Verma 2016406	Project Manager,Database Management(Web Developement Module)	

Juneesh Pawar E 20164003	Web Application Coding,Debugging	
Jyotirmoy Barman 20164139	Designing,Maintenance	
Kadamb Agarwal 20164107	Testing,Requirement gathering	
Kadoo Anagha Anil 20164033	Coding(Complexity evaluation module),Analysis	

10) References:

1. Radon 2.4.0 Documentation(https://radon.readthedocs.io/en/latest/index.html)
2. Article on Software Design Complexity(https://www.tutorialspoint.com/software_engineering/software_design_complexity.htm)
3. Python Documentation (https://www.python.org/doc/versions/)

11) Future Scope:

1. The project can be further expanded to include different languages i.e. more than one language.
2. Also the project can be further converted to a plugin for different IDE.