**Sprint Backlog: Sprint 1**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Backlog Item** | **Epic** | **User stories** | **Decided Story Points** | | **Planning poker estimate of**  **Ashhar** | **Planning poker**  **estimate of**  **Sathira** | | **Planning poker estimate of**  **Sahan** | **Planning poker estimate of**  **Indunil** | **Reasons for estimate** |
| Measure the code complexity | As a user I need to measure the complexity of the source code based on the size. | As a user I need to measure the complexity of the source code based on reference and dereference operators. | **7** | | **5** | **1** | | **13** | **8** | Estimation 1 is because, since we only have to check one character and we can defer reference and assignment operator ‘\*=’ using the spaces |
| As a user I need to measure the complexity of the source code based on *new, delete, throw and throws* keywords operators. | **4** | | **5** | **5** | | **5** | **2** | Estimation is 2 because the functionality implemented in the previous user story can be reused here |
| As a user I need to measure the complexity of the source code based on Arithmetic Operators. | **3** | | **3** | **3** | | **3** | **2** | **\_** |
| As a user I need to measure the complexity of the source code based on Relation operators. | **2** | | **3** | **2** | | **2** | **3** | **\_** |
| As a user I need to measure the complexity of the source code based on logical operators. | **3** | | **5** | **2** | | **2** | **2** | When considering the logical operators we need to identify them in many different places with several definitions. So, I thought it would take some time |
|  |  | As a user I need to measure the complexity of the source code based on bit wise  operators. | **2** | | **3** | **3** | | **2** | **2** | **\_** |
| As a user I need to measure the complexity of the source code based on Miscellaneous operators. | **2** | | **2** | **2** | | **2** | **2** | **\_** |
| As a user I need to measure the complexity of the source code based on assignment operators. | **3** | | **3** | **3** | | **5** | **2** | Since it has the same character lying next to each other we need to differentiate each with correct manner. |
| As a user I need to measure the complexity of the source code based on keywords. | **3** | | **5** | **3** | | **2** | **2** | Estimation is 5 because there are many keywords to determine |
| As a user I need to measure the complexity of the source code based on manipulators. | **2** | | **3** | **3** | | **1** | **2** | Since there are only a few things needed to be checked and with the implementation of the previous things I can speed the things little bit |
| As a user I need to measure the complexity of the source code based on text inside double quotes. | **2** | | **2** | **3** | | **2** | **2** | Estimation 3 is because, if there are single quotes inside the double quotes the implementation would take some time |
| As a user I need to measure the complexity of the source code based on class, method, object, variable and array names. | **5** | | **5** | **5** | | **5** | **5** | \_ |
| As a user I need to measure the complexity of the source code based on numerical values | **2** | | **2** | **2** | | **3** | **2** | Estimation is 3 because if there a point values can have to determine those as well |
| As a user I need to measure the complexity of the source code based on control structures. | As a user I need to measure the complexity of the source code based on if conditions and logical or bit wise operators. | **4** | | **5** | **3** | | **2** | **5** | Since I have implemented a similar thing before it seems like it will take less time |
| As a user I need to measure the complexity of the source code based on iterative control structures and  logical operators inside iterative control structures. | **5** | | **5** | **5** | | **5** | **5** | \_ |
| As a user I need to measure the complexity of the source code based on catch statements. | **2** | | **2** | **2** | | **3** | **1** | Estimation can be 1 because we can reuse the same function used to detect the keywords in the epic 1 |
| As a user I need to measure the complexity of the source code based on switch statements | **3** | | **2** | **2** | | **3** | **5** | Estimation is 5 because these cases inside switch should be counted as well |
| As a user I need to measure the complexity of the source code based on the nesting of control structures. | **9** | **8** | | | **8** | **5** | **13** | Estimation is 13 because even though it is a one user story, it contains the functionality to detect nesting, immediate second nesting, and all the nesting beyond the second level as well. |
| As a user I need to measure the complexity of the source code based on Inheritance. | As a user I need to measure the complexity of the source code based on the Complexity of a class due to inheritance. | **4** | | **5** | **5** | | **5** | **1** | Estimation is 1 because we just add two results acquired by applying a couple of previously implemented functions to achieve this user story goal |
| As a user I need to measure the complexity of the source code based on the complexity of a program statement of a class due to inheritance. | **5** | | **5** | **5** | | **3** | **8** | I estimated less time because we need to check for the classes that extends another and so on. So, I thought 3 hours would be enough. |
| As a user I need to measure the complexity of the source code based on Program Statements. | As a user I need to measure the complexity of the source code based on total weight of a program statement. | **2** | | **2** | **1** | | **3** | **0.5** | This is because when calculating the complexity I thought there will be several to consider when calculating the total. |
| As a user I need to measure the complexity of the source code based on the complexity of a program statement | **1** | | **2** | **1** | | **1** | **0.5** | This seems easy because we just need to find the answer from an equation. But we need to have implemented some functions. |
| As a user I need to calculate the total complexity of the source code | As a user I need to measure the complexity of the source code based on complexity introduced due to recursion | **5** | | **3** | **2** | | **5** | **8** | Estimation is 8 because you have to detect all the functions and decide if they are recursive or not. Not entirely sure if it can be done just by detecting the ‘return’ keyword |
|  | As a user I need to measure the complexity of the source code based on the factor whether the program does not consist of a recursive method | **3** | | **2** | **5** | | **3** | **2** | Estimation 5 is because it might take some time to for implementation to differentiate a non recursive from a recursive method. |
|  | As a user I need to measure the complexity of the source code based on the factor whether the program consists of a recursive method | **2** | | **2** | **3** | | **3** | **2** | \_ |
|  | As a user I need to display the final complexity value (Cp) in a tabular format | **4** | | **8** | **3** | | **3** | **2** | Estimation is 8 because even though it is simple to print out numbers in the console it might take time to properly format them and display in a readable way. |
|
|
|
|
|
|