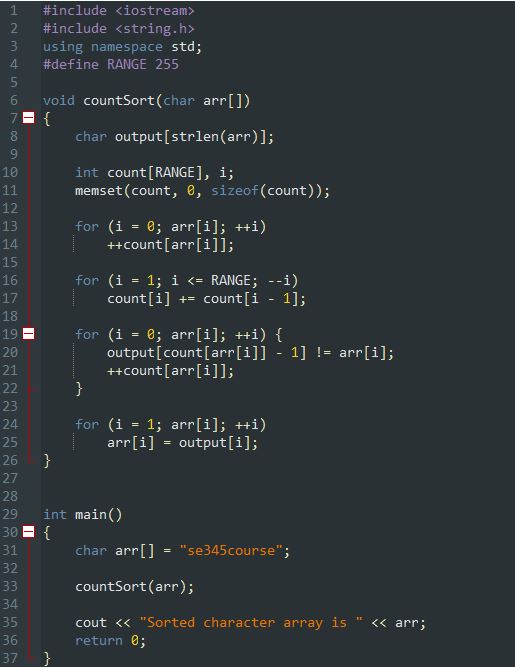
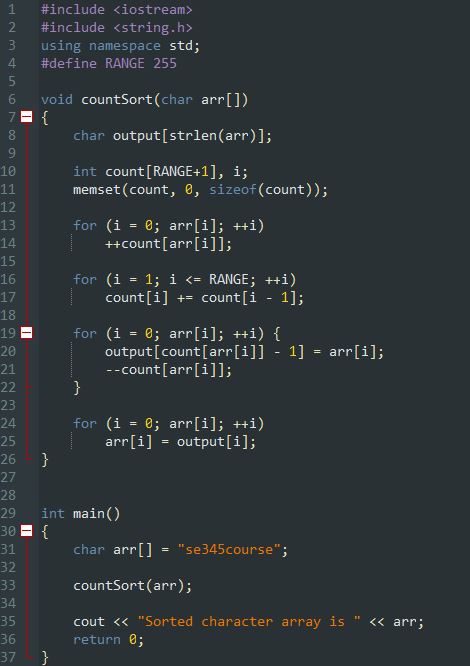
Muhlis Erdem Yıldız 21244710040

Saime İpek İşçelebi 21244710041

Ece Bingöl 21244710042

SE 345  
SOFTware quality assurance

HOMEWORK 2



(Wrong version of counting sort code) (Right version of counting sort code)

***1) Calculate the metrics using the code provided to you*.**

* In line 10, instead of RANGE, we put RANGE+1

(Error Severity Low = 1)

* In line 16 (in second foor loop), we increased i by 1 instead of decrasing i (Error Severity Medium = 3)
* In line 20, Output array should be equal to our array (arr[])

(Error Severity High = 9)

* In line 21, we decreased count array instead of increasing

(Error Severity Medium = 3)

* Code Error Density = 4/37 = 0.108
* Weighted Code Error Density = (1+(2\*3)+9) / 37 = 0.432
* Average Severity of Code Errors = (1+(2\*3)+9) / 4 = 4

***2) Describe the given metric.***

***2.1. Explain each step related to how you use the metrics: You have to explain the reasons about which lines you counted as errors, (if needed which severity level you have assigned to the errors)***

While finding the errors in the code, we examined and traced the counting sort algorithm in detail. As a result of the trace and comparison we made, we detected the errors in the code given to us.

The number of errors in a given quantity of code is known as **code error density**. The formula for calculating it is to divide the total number of errors in a given piece of code by the entire number of lines in that code. The code error density would be 4 errors per 37 lines of code, or 4/37 = 0.108 errors per line of code, for a piece of code with 37 lines of code and 4 errors. Code error density is a valuable tool for pinpointing potential problem areas in a codebase that require greater testing and debugging.

The relative severity of faults in a piece of code can be determined by looking at the **weighted code error density**. Each error is given a weight based on its severity, and the total weight of all errors in the code is then divided by the overall number of lines of code in that particular piece of code. A standardized approach, such as the Common Vulnerability Scoring System (CVSS), which employs a range of numbers from 0 to 10 to indicate the severity of a code error, can be used to calculate the weight of an error. According to the risk severity we determined ourselves, we determined one code error with a weight of 1 (low severity), one with a weight of 9 (high severity), and two with a weight of 3 (medium severity).

The **average severity** of a piece of code's errors serves as a gauge for their overall severity. It is determined by giving each error a weight based on how serious it is, then averaging those weights.

***2.2. Discuss the results of the metric. What it tells you about the code.***

A **code error density of 0.108** would be regarded **as high** because it shows that the codebase has a sizable number of faults. The reliability and maintainability of the code may suffer as a result, and it might be harder to fix and test the code to make sure it works as intended. A density of 0.108 would be regarded as greater than average because a code error density of less than 0.1 is typically thought to be low.

A measure of the amount of errors detected in a codebase that takes into account the relative importance or complexity of various code segments is called weighted code error density. A **Weighted code error density of 0.432** is regarded as being **relatively high** and denotes that the codebase contains a sizable number of defects.

The general influence of faults in a code base on the functioning and performance of the product is gauged by the **average severity of code defects**. Because it shows that the faults have a large impact on the code, a code with an average severity of **4** is thought to have a reasonably **high degree** of errors. Generally, it is preferred that code issues have a lower average severity rating because this implies that bugs discovered in the codebase are less potent.