## Review Form

**Authors’s Name:\_\_Ziyu ZENG\_ Author’s Student No.: \_\_2019011343\_**

**Reviewer’s Name:\_\_Wuwei\_YUAN\_\_ Reviewer’s Student No.: \_\_2020040054\_ Date：03-09**

|  |  |
| --- | --- |
| **Makefile**  **(10%)** | 8 |
| **Comments** | Problem1: The extension of the output file is .exe  Problem2: Missing makefile  Problem3: Missing makefile |
| **Compilation**  **(编译正确) (10%)** | 10 |
| **Comments** | Compile successfully. |
| **Correctness of Results**  **(结果正确) (40%)** | 35 |
| **Comments** | Problem1: The result of the posynomial is wrong. (It is the same as the polynomial.)  Problem2: You should choose 3 reviewers for each student on the student list instead of only one student. |
| **Naming Convention**  **(变量命名合理) (5%)** | 4 |
| **Comments** | Uses too much abbreviation. This may confuse when others reading your code. |
| **Code Formatting**  **(代码格式合理) (5%)** | 4 |
| **Comments** | Using a good coding style can make your code more clear. |
| **Code Comments**  **(代码注释合格) (5%)** | 4 |
| **Comments** | Lack of descriptions of classes and functions. |
| **Other Coding Style and efficiency** (代码运行效率) **(10%)** | 10 |
| **Comments** | Overall, it’s fast. |
| **OOP Design Style**  **(15%)** | 13 |
| **Comments** | Problem1: I think the for-loop for computing many times should be placed outside the computing function because the function of that function is only computing the polynomial.  Problem2: The fixed number of reviewers and the length of arrays makes it difficult to reuse when changing the number of students and reviewers.  Problem3: You should select the slowest algorithm as the baseline in your code even though the bubble sort is always the slowest. |
| **Total Score (0-100)** | 88 |
| **Overall Comments** | Good job, but there is still a lot of room for improvement.  You should try Google Coding Style. |
| **Version of Reviewer’s compiler** | GNU Make 4.1  g++ (Ubuntu 9.2.1-17ubuntu1~18.04.1) 9.2.1 20191102 |

**NOTE:**

1. For coding styles including naming, formatting, code comments, etc., please refer to the Google C++ Style Guide. You may stick to your own coding style if you already have one, provided that your coding style is easy to read and understand by others.

(<https://google.github.io/styleguide/cppguide.html>)

2. OOP design: easy for code reuse (代码复用); easy for extension and adaptability to future change for new user requirements (易于扩展，适应未来用户需求的改变). We will learn these OOP design features throughout this semester.

3. The reviewer (评阅人) is responsible for filling in the review form with both credits (分数) and detailed comments/suggestions (评价/建议) for further improvements (改进), and then returns the review form to the author of the code.

4. Finally, each student needs to submit the improved (改进后的) source code according to the comments, as well as a .doc or .pdf file including the returned review forms and statements on the revised code in reply to the reviewers’ comments.

5. Please comply with our “**Rules for Submission**” when submitting your homework.