Basics 5 – Debugging C++

Student Information

Integrity Policy: All university integrity and class syllabus policies have been followed. I have neither given, nor received, nor have I tolerated others' use of unauthorized aid.

I understand and followed these policies: (•) Yes ( ) No

Name: Joey Domino

Date: 10/13/2021

Submission Details

Final ***Changelist*** number: 117647

Verified build: (•) Yes ( ) No

Number Tests Passed: 10

Required Configurations: DEBUG only

Discussion (What did you learn):

I learned:
- How to catch some really annoying bugs.

Verify Builds

- Follow the Piazza procedure on submission
  - Verify your submission compiles and works at the changelist number.
- Verify that only MINIMUM files are submitted
  - No – Generated files
    - *.pdb, *.suo, *.sdf, *.user, *.obj, *.exe, *.log, *.pdb, *.db, *.user
    - Anything that is generated by the compiler should not be included
  - No – Generated directories
    - /Debug, /Release, /Log, /ipch, /.vs
- Typical files project files that are required
  - *.sln, *.cpp, *.h
  - *.vcxproj, *.vcxproj.filters, CleanMe.bat

Standard Rules

Submit multiple times to Perforce

- Submit your work as you go to perforce several times (at least 5)
  - As soon as you get something working, submit to perforce
  - Have reasonable check-in comments
    - Points will be deducted if minimum is not reached

Write all programs in cross-platform C++

- Optimize for execution speed and robustness
- Working code doesn't mean full credit

Submission Report

- Fill out the submission Report
  - No report, no grade

Code and project needs to compile and run

- Make sure that your program compiles and runs
  - Warning level ALL …
  - NO Warnings or ERRORS
    - Your code should be squeaky clean.
  - Code needs to work "as-is".
    - No modifications to files or deleting files necessary to compile or run.
  - All your code must compile from perforce with no modifications.
    - Otherwise it's a 0, no exceptions

Project needs to run to completion

- If it crashes for any reason…
  - It will not be graded and you get a 0

No Containers

- NO STL allowed {Vector, Lists, Sets, etc...}
  - No automatic containers or arrays
  - You need to do this the old fashion way - ***YOU EARNED IT***

Leave Project Settings

- Do NOT change the project or warning level
  - Any changing of level or suppression of warnings is an integrity issue

Simple C++

- No modern C++
  - No Lambdas, Autos, templates, etc…
  - No Boost
- NO Streams
  - Used fopen, fread, fwrite…
- No code in MACROS
  - Code needs to be in cpp files to see and debug it easy
- ***Exception:***
  - implicit problem needs templates

Leaking Memory

- If the program leaks memory
  - There is a deduction of 20% of grade
- If a class creates an object using new/malloc
  - It is responsible for its deletion
- Any ***MEMORY*** dynamically allocated that isn't freed up is ***LEAKING***
  - Leaking is ***HORRIBLE***, so you lose points

No Debug code or files disabled

- Make sure the program is returned to the original state
  - If you added debug code, please return to original state
- If you disabled file, you need to re-enable the files
  - All files must be active to get credit.
  - Better to lose points for unit tests than to disable and lose all points

No Adding files to this project

- This project will work “as-is” do not add files…
- Grading system will overwrite project settings and will ignore any student’s added files and will returned program to the original state

UnitTestConfiguration file (if provided) needs to be set by user

- Grading will be on the UnitTestConfiguration settings
  - Please explicitly set which tests you want graded… no regrading if set incorrectly

Due Dates

- See Piazza for due date and time
- Submit program perforce in your student directory assignment supplied.
- Fill out your this ***Submission Report*** and commit to perforce
    - ***ONLY*** use Adobe Reader to fill out form, all others will be rejected.
    - Fill out the form and discussion for full credit.

Goals

- C++ Proficiency
    - Real-world debugging and troubleshooting
    - Increasing C++ knowledge and understanding

Assignments

- General:
    - Refactor all programs to work correctly in Basics5 solution.
        - The expected answers are in each problem.
        - Once you fixed each program
            - Run the supplied batch file test
        - All programs should run and execute without hanging with correct answers
        - Verify for each program…
            - ***Use Start without Debugging <cntrl>+F5***
    - Follow the instructions for each problem
        - Every problem has typically a one line or one method fix to make the given problem work as intended.

- Background
    - Each program has some error, but it's not obvious
        - All programs should compile with Warning level=ALL with no compile errors or warnings
    - The big question you will be working on,
        - "Why doesn't my program work correctly?"
    - You will need to look at reference material (books, lectures, etc.) to understand why the code is behaving is an unexpected way.
    - You will need to step through the code, add break points and print statements to help you see what is going on wrong.
    - DO NOT CHANGE the "focus" of the program to get around the error.
        - Instead, leave it and fix the subtle error.
        - If you rework the complete problem instead of fixing the error → you are not learning the exercise.

- Use the solution file.
  - You need to select an individual solution to start the program
    - For example, project A
    - Start – Basics5_A.sln
  - Ask questions in Piazza if confused.
    - → ***DO NOT POST THE ANSWER***
    - ***Each fix is only 1 or 2 lines… don't give away the answers***

- Issues
  - Sometimes it works as you single step, but if you run the program as
    - Start without Debugging <Ctrl>+F5 – you see a crash or an error
    - Yes, that's a bug!
      - " What? How can that be? "
    - Try to fix, the debugger might be hiding something from you.
      - Dig deep figure it out.
      - Debuggers often do a lot of things that we are not aware of...
        - Start a Piazza thread with your discoveries

- Testing / Verification
  - I supplied the test scripts
    - Only run this AFTER you fixed all 10 projects
  - I will use for these scripts for the grading
    1. Compile the solution first (build all the projects)
    2. RunDebugTests.bat
  - Those functions generate a single Output.txt to view the complete output in each of the respective directories.
  - They are there to try,
    - As is, there are errors,
    - Just hit abort or cancel many times to pound through the script.
    - look at Output.txt for results

- This is for you, do your own work!
  - Feel free to talk with others about setup, version control
    - ***DO NOT SHARE*** coding ideas on this assignment
  - Do not copy your friend's code.
    - This is a competition!
  - If you don't understand....
    - Piazza, piazza, piazza
  - These basics is designed to shake out some understanding that you do or don't know. So it personal, each person will have their own issues...
    - Sharing doesn't help YOU!

Validation

Simple checklist to make sure that everything is submitted correctly

- Is the project compiling and running without any errors or warnings?
- Did you do all 10 problems?
- Does the project run ***ALL*** the unit tests execute without crashing?
- Is the submission report filled in and submitted to perforce?
- Follow the verification process for perforce
    - Is all the code there and compiles “as-is”?
    - No extra files
- Is the project leaking memory?

Hints

Most assignments will have hints in a section like this.

- Do many little check-ins
    - Iteration is easy and it helps.
    - Perforce is good at it.
- Look at the C++ programming language, ***Effective C++*** books
    - A lot of good ideas in there.
- Step through this code, line by line
    - Then you can see the mistakes