CSE 231 Problem Set 07

# Problem 35.1: Crash in Library

Consider the following scenario:

Your program crashed, which halted in the debugger. Unfortunately, the crash happened in a generic library function used in hundreds of places in the program. You would like to know which line of code in your program called this library function.

Identify which debugger tool(s) would be helpful in diagnosing the problem. Briefly explain why a given tool would not be useful or, if it would be useful, how would you use it.

|  |  |
| --- | --- |
| Run |  |
| Step-Over |  |
| Step-Into |  |
| Step-Out |  |
| Run-To |  |
| Set-Next-Statement |  |
| Watch |  |
| Call stack |  |
| Disassembly |  |
| Standard Breakpoint |  |
| Conditional Breakpoint |  |

# Problem 35.2: Polymorphic Variable

Consider the following scenario:

You have the program halted in a function that works with a polymorphic variable. It would really help to know which derived class was associated with the variable you are looking at right now.

Identify which debugger tool(s) would be helpful in diagnosing the problem. Briefly explain why a given tool would not be useful or, if it would be useful, how would you use it.

|  |  |
| --- | --- |
| Run |  |
| Step-Over |  |
| Step-Into |  |
| Step-Out |  |
| Run-To |  |
| Set-Next-Statement |  |
| Watch |  |
| Call stack |  |
| Disassembly |  |
| Standard Breakpoint |  |
| Conditional Breakpoint |  |

# Problem 35.3: Blocking Bug

Consider the following scenario:

It took you 10 minutes to get the program in a right state where the program exhibits the bug. Unfortunately, a simple error in the code is preventing you from moving on and discovering the real problem. You really don’t want to recompile and spend another 10 minutes to get to the same spot.

Identify which debugger tool(s) would be helpful in diagnosing the problem. Briefly explain why a given tool would not be useful or, if it would be useful, how would you use it.

|  |  |
| --- | --- |
| Run |  |
| Step-Over |  |
| Step-Into |  |
| Step-Out |  |
| Run-To |  |
| Set-Next-Statement |  |
| Watch |  |
| Call stack |  |
| Disassembly |  |
| Standard Breakpoint |  |
| Conditional Breakpoint |  |

# Problem 35.4: Reading a File

Consider the following scenario:

There is a problem in the file-reading code in your application. After reading about 100 records from a file, the subsequent ones are corrupt. You don’t want to use step-over 100 times! That will take forever.

Identify which debugger tool(s) would be helpful in diagnosing the problem. Briefly explain why a given tool would not be useful or, if it would be useful, how would you use it.

|  |  |
| --- | --- |
| Run |  |
| Step-Over |  |
| Step-Into |  |
| Step-Out |  |
| Run-To |  |
| Set-Next-Statement |  |
| Watch |  |
| Call stack |  |
| Disassembly |  |
| Standard Breakpoint |  |
| Conditional Breakpoint |  |

Copy the following C++ program into your debugger.

#include <iostream>

using namespace std;

int main()

{

int number;

int value;

// prompt

cout << "Enter a number: ";

cin >> number;

// first switch statement

switch (number)

{

case 1:

value = 0x101; break;

case 2:

value = 0x102; break;

case 3:

value = 0x104; break;

case 4:

value = 0x104; break;

case 5:

value = 0x105; break;

case 6:

value = 0x106; break;

case 7:

value = 0x107; break;

case 8:

value = 0x108; break;

case 9:

value = 0x109; break;

}

// second switch statement

switch (number)

{

case 1:

value = 0x101; break;

case 10:

value = 0x102; break;

case 100:

value = 0x104; break;

case 1000:

value = 0x104; break;

case 10000:

value = 0x105; break;

case 100000:

value = 0x106; break;

case 1000000:

value = 0x107; break;

case 10000000:

value = 0x108; break;

case 100000000:

value = 0x109; break;

}

// silly output

cout << value << endl;

return 0;

}

# Problem 35.5a: Disassembly

Execute the program on the previous page and break into the debugger. Check the disassembly for this code. How does your compiler treat the first SWITCH/CASE statement?

Paste in the first 20 lines of the disassembly:

Please read the Chapter 07 section on “Many Options” and see if you can classify the implementation of each as IF/ELSE-IF, Iteration, Binary Search, or Jump Table. Justify your answer.

# Problem 35.5b: Disassembly

Execute the program on the previous page and break into the debugger. Check the disassembly for this code. How does your compiler treat the second SWITCH/CASE statement?

Paste in the first 20 lines of the disassembly:

Please read the Chapter 07 section on “Many Options” and see if you can classify the implementation of each as IF/ELSE-IF, Iteration, Binary Search, or Jump Table. Justify your answer.