SWE301

Software Verification and Validation

Software Error Mitigation - Lab 2

**GDB Debugger**

**Tool:** [**https://www.onlinegdb.com/**](https://www.onlinegdb.com/)

**Help Guide:** [**https://www.onlinegdb.com/blog/brief-guide-on-how-to-use-onlinegdb-debugger/**](https://www.onlinegdb.com/blog/brief-guide-on-how-to-use-onlinegdb-debugger/)

|  |  |
| --- | --- |
| **Command** | **Description** |
| **run or r** | Executes the program from start to end. |
| **break or b** | Sets a breakpoint on a particular line. |
| **disable** | Disables a breakpoint |
| **enable** | Enables a disabled breakpoint. |
| **next or n** | Executes the next line of code without diving into functions. |
| **step** | Goes to the next instruction, diving into the function. |
| **list or l** | Displays the code. |
| **print or p** | Displays the value of a variable. |
| **quit or q** | Exits out of GDB. |
| **clear** | Clears all breakpoints. |
| **continue** | Continues normal execution |

Use GDB Debugger

Example 1:

#include <iostream>

#include <bits/stdc++.h>

using namespace std;

int main()

{

    int n = 9, div = 0;

div = n/0;

   cout << "result = "<< div;

}

Example 2:

#include<iostream>

int sum,sub,mult,div,module;

int a,b;

cout<<"Enter value of a ="

cin<<a;

cout<<"Enter value of b =;

cin<<b;

sum=a+b;

sub=a-bmult=a\*b;

div=a/'b';

module=a%b;

Practice 1:

1. Use Dynamic Analysis to debug the code. Share what did you do?

#include <stdio.h>

#include <string.h>

int main(int argc, char\* argv[]) {

/\* [1] \*/ char buf[256];

/\* [2] \*/ strcpy(buf,argv[1]);

/\* [3] \*/ printf("Input:%s\n",buf);

return 0;

}

Practice 2:

There is an issue with the Login & Register code. Resolve it.

1. Copy the code and compile
2. Debug using GDB Debugger (Dynamic Analysis)

Explain your Dynamic Analysis

(What did you do? What was the output? Did you manage to resolve the issue?)

Having to use both PVS Studio and GDB Debugger, how do these two tools complement each other?

Upload your answer in Moodle.