CS270 Recitation 3
"C Debugging Exercise"

Goals

To learn the gdb debugging tool and practice using it on a C program.

Instructions

Make a subdirectory called R3 for the recitation assignment, all files should reside in this subdirectory.
Copy the code shown below into a file called r3.c in your R3 subdirectory:

```
01: #include <math.h>
02:
03: // Function: quadratic
04: // Description: implements the quadratic equation
05: // Parameters: int, int, int: coefficients, float *, float *, pointer to roots
06: // Return: void
07: // Error Avoid division by zero
08: void quadratic(int coeff1, int coeff2, int coeff3, float *root1, float *root2)
09: {
10:    if (coeff1 == 0)
11:    {
12:        // Avoid division by zero
13:        *root1 = 0.0;
14:        *root2 = 0.0;
15:    }
16:    else
17:    {
18:        // Implement quadratic equation
19:        *root1=(-coeff2+sqrt((coeff2*coeff2)-(4*coeff1*coeff3)))/(2*coeff1);
20:        *root2=(-coeff2-sqrt((coeff2*coeff2)-(4*coeff1*coeff3)))/(2*coeff1);
21:    }
22: }
```

Copy the code shown below into a file called main.c in your R3 subdirectory:

```
01: #include <stdio.h>
02:
03: // Function declaration
04: void quadratic(int coeff1, int coeff2, int coeff3, float *root1, float *root2);
05:
06: // Program entry point
07: int main()
08: {
09:    int a, b, c;
10:    float r1, r2;
11:
12:    printf ("Quadratic Program\n");
13:    printf("Enter a: ");
14:    scanf("%d", &a);
15:    printf("Enter b: ");
16:    scanf("%d", &b);
17:    printf("Enter c: ");
18:    scanf("%d", &c);
19:    quadratic(a, b, c, &r1, &r2);
20:    printf("Roots are %3.2f and %3.2f\n", r1, r2);
21: }
```

Compile the program into an executable called r3, as shown below.

gcc -g -std=c99 -Wall -c r3.c -o r3.o
gcc -g -std=c99 -Wall -c main.c -o main.o
gcc -g -lm r3.o main.o -o r3

To debug the compiled program, type the following command:

```
$ gdb r3                    // start gdb debugger
```

Use the following debugger commands to run the program and examine variables:

```
(gdb) set logging on        // enable logging to gdb.txt
(gdb) list 20               // listing around line 20
(gdb) break 19              // set breakpoint at line 19 in main.c
(gdb) break 20              // set breakpoint at line 20 in main.c
(gdb) run                   // run program
```

Enter integer values for a, b, and c as requested by the program.

```
(gdb) print a               // print value of a
(gdb) print b               // print value of b
(gdb) print c               // print value of c
(gdb) print r1              // print value of r1
(gdb) print r2              // print value of r2
(gdb) print &r1             // print address of r1
(gdb) print &r2             // print address of r2
(gdb) step                  // step one line
(gdb) break 21              // set breakpoint at line 21 in r3.c
(gdb) continue              // continue to breakpoint just set
(gdb) print root1           // print value of address in root1
(gdb) print root2           // print value of address in root2
(gdb) print *root1          // print value of variable pointed at by root1
(gdb) print *root2          // print value of variable pointed at by root2
(gdb) x root1               // examine memory at address pointed to by root1 (hex)
(gdb) x root2               // examine memory at address pointed to by root1 (hex)
(gdb) x /f root1            // examine memory at address pointed to by root1 (float)
(gdb) x /f root2            // examine memory at address pointed to by root1 (float)
(gdb) info breakpoints      // list all breakpoints set above
(gdb) disable 1             // disable first breakpoint
(gdb) info breakpoints      // list all breakpoints set above
(gdb) enable 1              // reenable first breakpoint
(gdb) delete 1              // delete first breakpoint
(gdb) info breakpoints      // list all breakpoints set above
(gdb) help                  // show help message command categories
(gdb) help status           // show help message for status commands
(gdb) info stack            // display stack status
(gdb) info functions        // display function status
(gdb) continue              // continue to last breakpoint
(gdb) info locals           // display local variables status
(gdb) quit                  // quit debugger
```

Display the output of your debugging session, show it to the teaching assistant, and submit the R3.txt file to the Recitation 3 drop box on RamCT.

```
$ cp gdb.txt R3.txt
$ gedit R3.txt
```

Challenge: Figure out how to display the contents of r1 and r2 every time the breakpoint at line 19 in main.c is hit, using the gdb ‘display’ command.