CST8234 – C Programming F17 (Lab 1b)

Using gdb

By the end of this lab, you should have tried running the GNU debugging environment (gdb).

Compile and run divide.c using gdb

1. Download divide.c

2. Compile the program to be able to use gdb

```
# gcc -g -o divide division.c -ansi -pedantic -Wall
```

3. Run gdb and load `divide`

```
# gdb
(gdb) file divide
Reading symbols from CST8234/Labs/00_Lab/divide...done.
```

4. Run `divide`:

You can use the run command or the short version **r**

```
(gdb) r
Starting program: CST8234/Labs/00_Lab/divide divide
10 / 5 = 2
Program received signal SIGFPE, Arithmetic exception.
0x00000000004005bb in divide (a=10, b=0) at division.c:47
47 return a / b;
```

gdb is saying that it encountered an arithmetic exception (SIGFPE) when running the program. The error was encountered at line 47 in the program division.c The program was executing the function divide with arguments a=10 and b=0

Line 47 executes return a / b;

5. Use the command **`list`** or **`l`** for short. This command allows you to see the context of the crash, listing the code near around line 47 of divide.c

```
(gdb) list
45 int divide( int a, int b ) {
46
47 return a / b;
48 }
```

6. Move one level of execution up with the **`up`** command. The crash happened at the `divide` function.

You want to see what happens in the function that called `divide`, in this case `main`.

```
(gdb) up
#1  0x0000000000400579 in main () at division.c:34
34 divide( x, y );
(gdb) list
29 y = 5;
30
31 printf("%d / %d = %d\n", x, y, divide( x, y ));
32
33 y = 0;
34 divide( x, y );
35 printf("%d / %d = %d\n", x, y, divide( x, y ));
36
37 return 0;
38 }
```

7. Print the values of the variables `x` and `y`. Notice that when in the `divide` function the arguments are called `a` and `b`

```
(gdb) print x
$1 = 10
(gdb) print y
$2 = 0
```