

# Programming BootCamp

Debugging C Applications



# Outline

## ❖ Introduction to Debugging

## ❖ Debugging Main Concepts

- Breakpoint, Step, Watch...etc.

## ❖ Debugging Tools

- GDB, CDB, LLDB...etc.

## ❖ Debugging Demos

- Using an IDE (Qt creator, VS Code...etc.)
- Using Console

## ❖ Common Bugs

- Overflow, Invalid Pointers, Access Violation, ...etc.

# Introduction

## ❖ **What is Debugging?**

The process of finding and fixing bugs in computer programs

## ❖ **What is a Software Bug?**

A software bug is a flaw in the computer code that causes the program to fail or produce wrong or unexpected results

## ❖ **Types of Bugs**

Typos, Mistakes, Logic flaws...etc.

# Debugging Process

## ❖ **Observe the bug**

First we notice that the program is not functioning properly

## ❖ **Reproduce the bug**

We determine a sequence of steps and/or data to reproduce the problem

## ❖ **Fix the bug**

We use debugging tools to track down and fix the problem

# Debugging Main Concepts

## ❖ **Breakpoint**

Pause/interrupt the program execution to allow the programmer to inspect current state of the program

## ❖ **Step**

Allow the programmer to control the execution of code

## ❖ **Watch**

Allows the programmer to inspect variables, evaluate expressions, and examine value changes

# Popular C/C++ Debuggers

Debugger	Platform	Developer
GDB	Cross-Platform	Open-Source
CDB	Windows	Microsoft
LLDB	Cross-Platform Popular for macOS	Open-Source, Part of LLVM



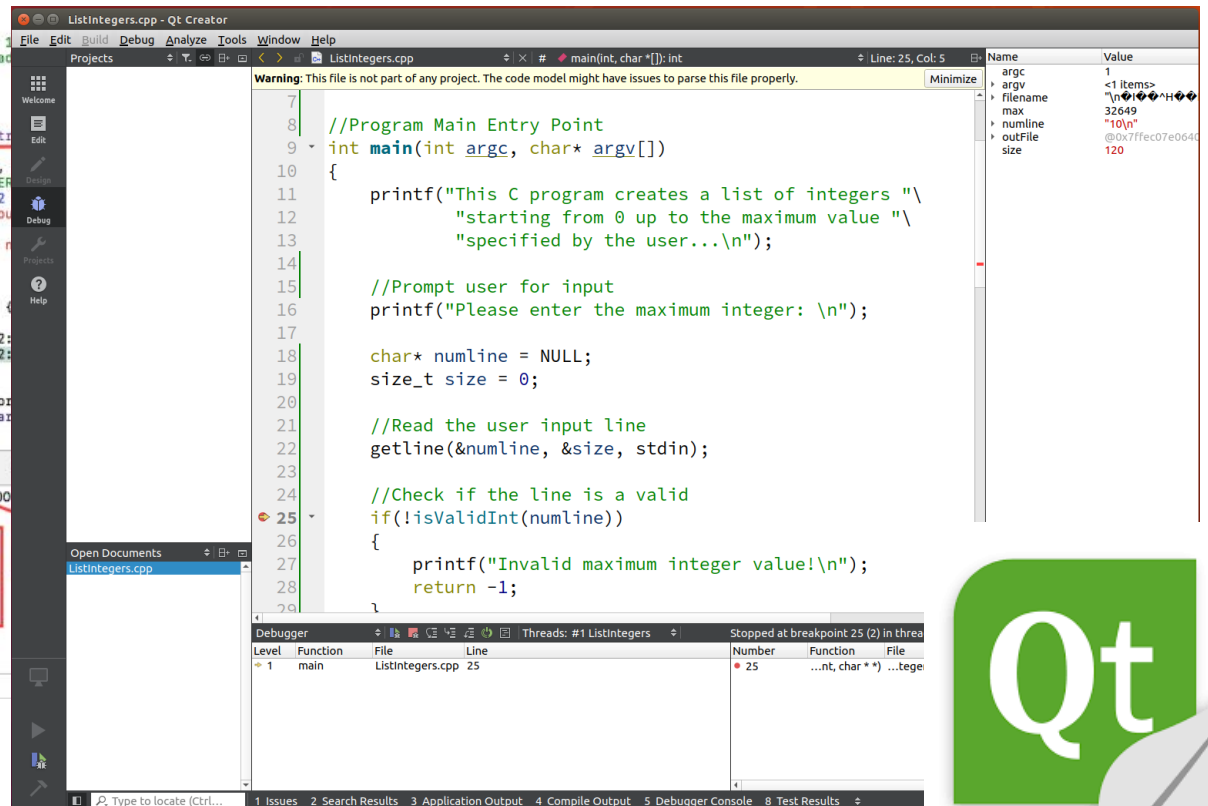
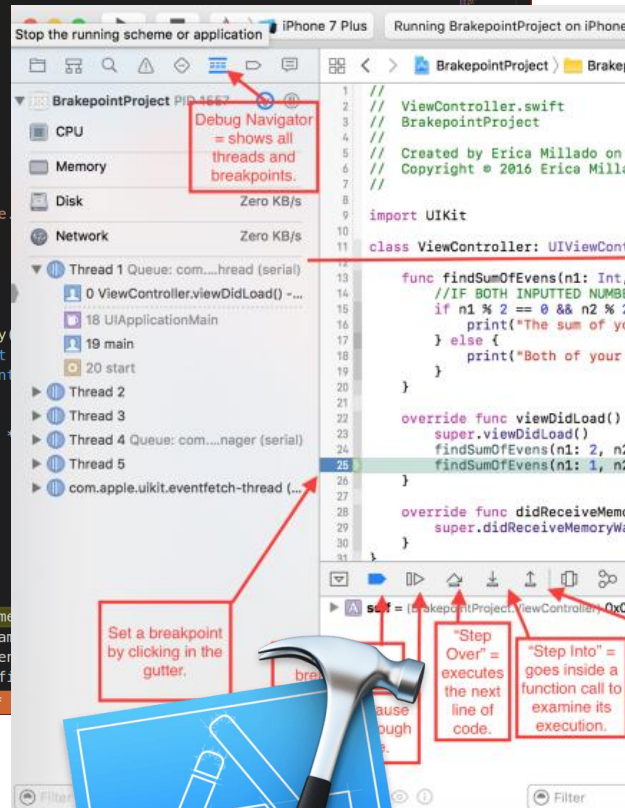
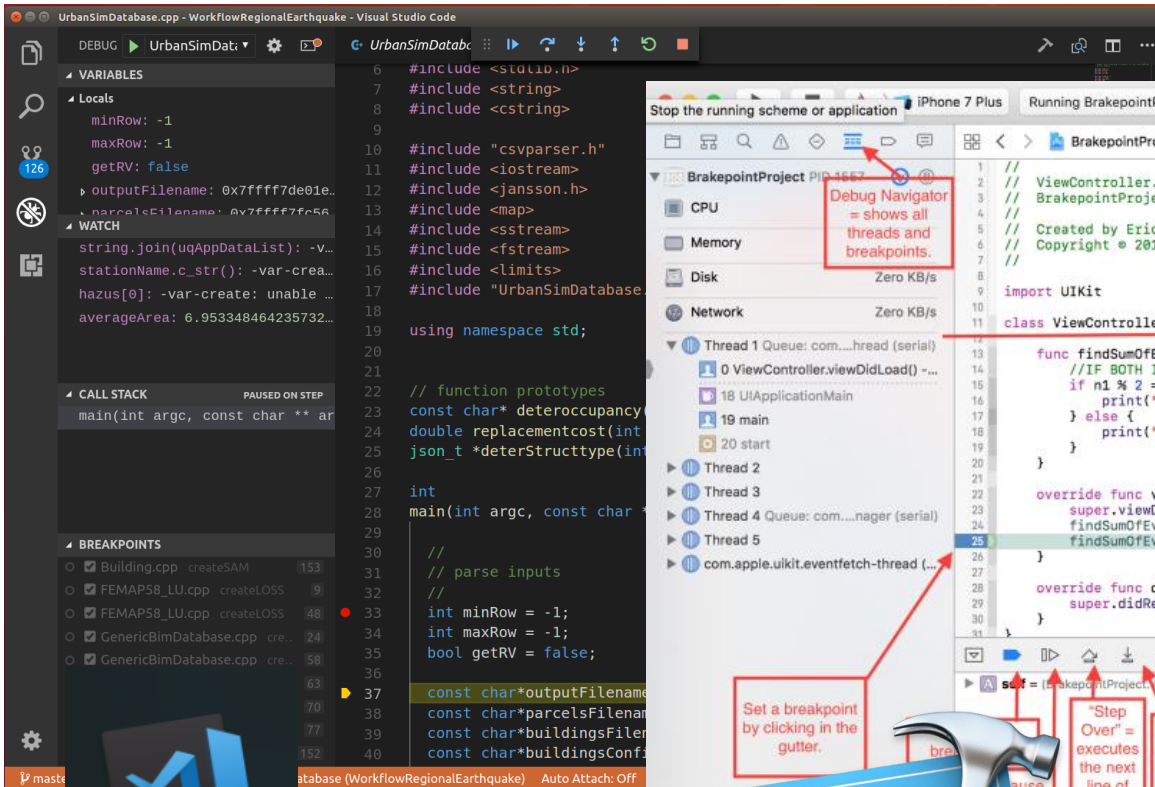
**GDB**  
The GNU Project  
Debugger

# Popular IDE Debugging Tools

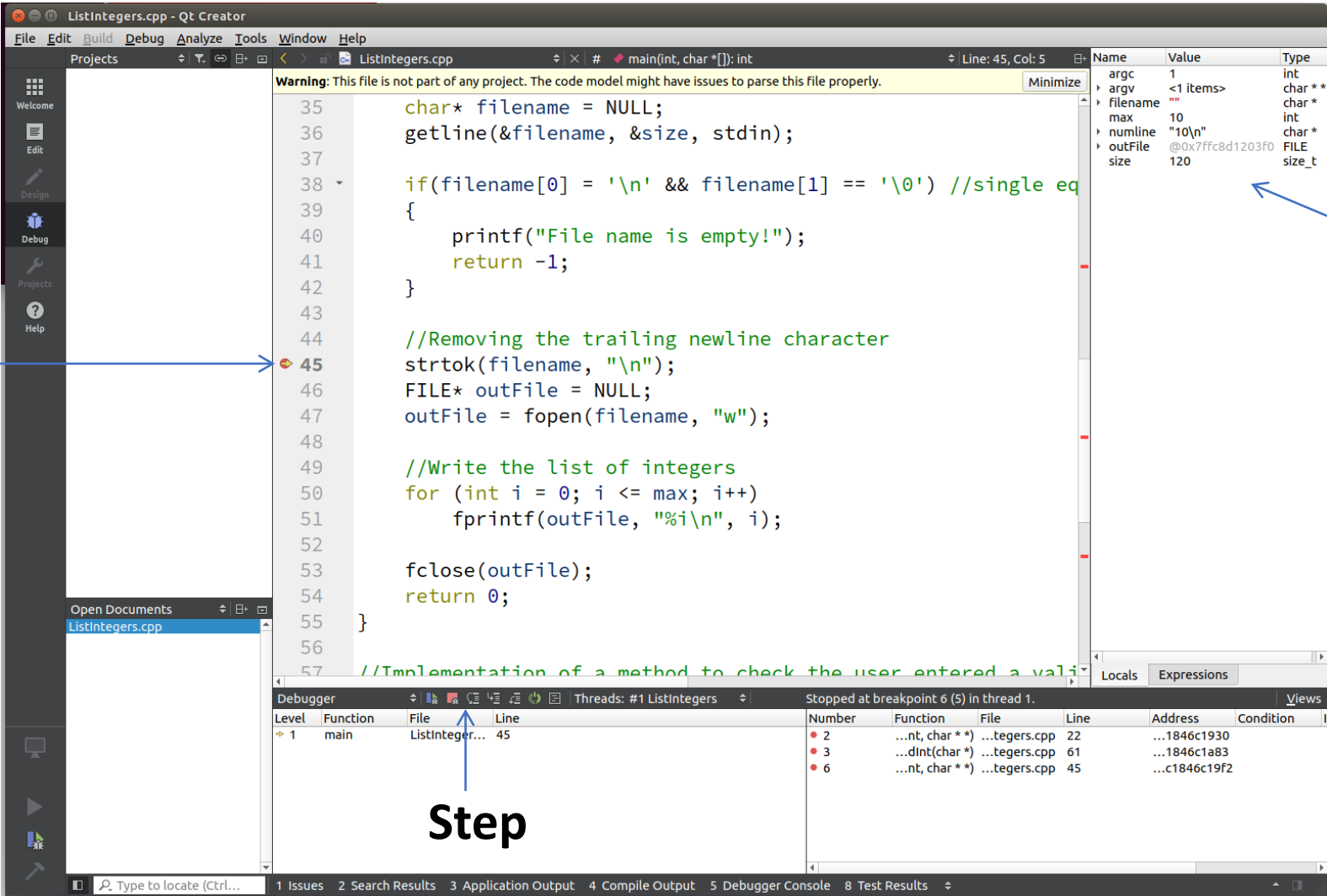
VS or VS Code

Xcode

Qt Creator



# Qt Creator Debugging Tools



The screenshot shows the Qt Creator IDE with a C++ file named `ListIntegers.cpp`. A breakpoint is set at line 45, which is highlighted with a red arrow and the label "Breakpoint". The code in the editor is as follows:

```

35 char* filename = NULL;
36 getline(&filename, &size, stdin);
37
38 if(filename[0] == '\n' && filename[1] == '\0') //single eq
39 {
40     printf("File name is empty!");
41     return -1;
42 }
43
44 //Removing the trailing newline character
45 strtok(filename, "\n");
46 FILE* outFile = NULL;
47 outFile = fopen(filename, "w");
48
49 //Write the list of integers
50 for (int i = 0; i <= max; i++)
51     fprintf(outFile, "%i\n", i);
52
53 fclose(outFile);
54 return 0;
55 }
56
57 //Implementation of a method to check the user entered a valid

```

The Watch window on the right shows the following variables and their values:

Name	Value	Type
argc	1	int
argv	<1 items>	char **
filename	""	char *
max	10	int
numline	"10\n"	char *
outFile	@0x7ffc8d1203f0	FILE
size	120	size_t

A blue arrow points from the "Watch" label to the Watch window. The Debugger window at the bottom shows the current state of the program, with a red arrow pointing to the "Step" button and the label "Step". The debugger console shows the following output:

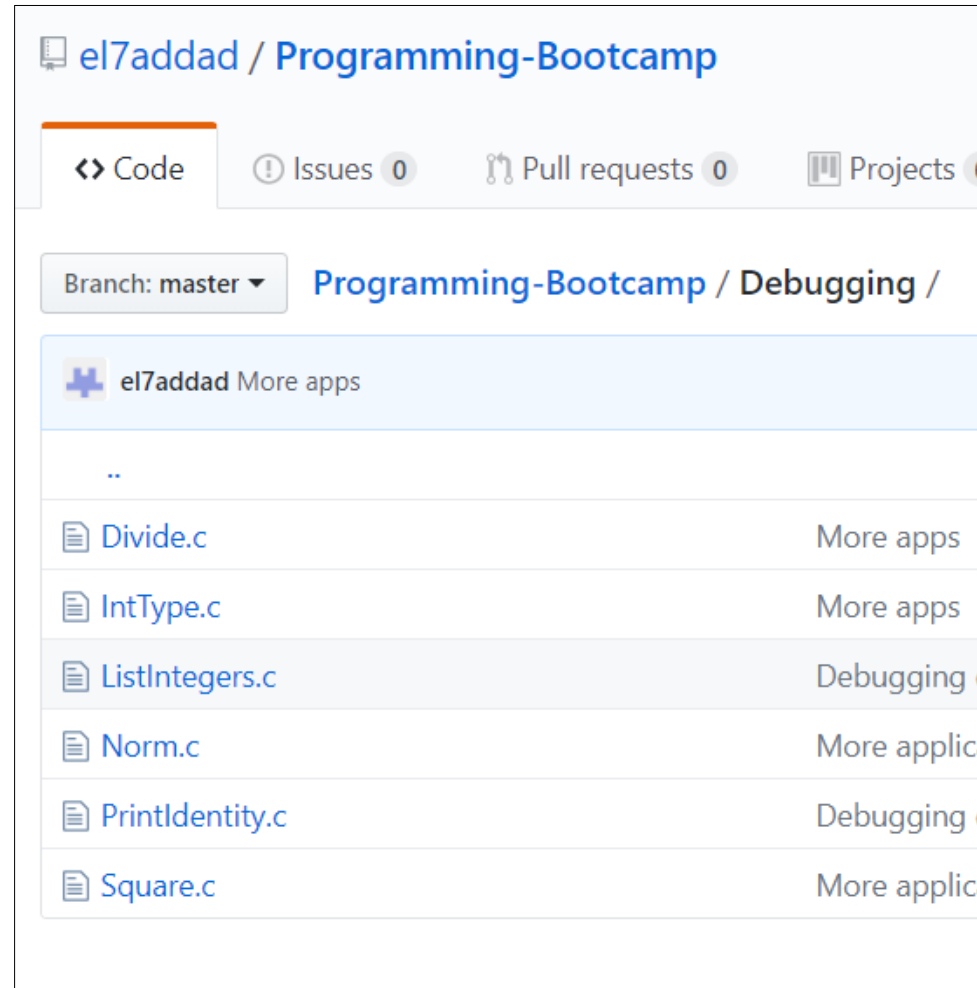
```

Level Function File Line
+ 1 main ListInteger... 45

```



# Debugging Demos



git clone <https://github.com/el7addad/Programming-Bootcamp.git>

# Debugging Demo: List Integers Application

## ❖ Compile the code

```
gcc -o ListIntegers ListIntegers.cpp
```

## ❖ Running the application

```
./ListIntegers
```

## ❖ Compile the code with debug info

```
gcc -g -o ListIntegers ListIntegers.cpp
```

# Debugging Demo: Print Identity Application

## ❖ Compile the code

```
gcc -o PrintIdentity PrintIdentity.cpp
```

## ❖ Running the application

```
./PrintIdentity
```

## ❖ Compile the code with debug info

```
gcc -g -o PrintIdentity PrintIdentity.cpp
```