

**Course:** ECE-T580 Compiler Design

**Date:** 09/23/2021

**Instructor:** Dr. Kandasamy

**Teaching Assistant:** Mr. Shihao Song

Ms. Arghavan Mohammadhassani

**Group Student:** Quoc Thinh Vo - 14419634

Tuong Tran – 14190713

## **Project: LLVM Tutorial**

### **I. Project implementation and output:**

The function `getInstructionCount()` was implemented to get the number of lines of instructions.

Code snippet:

```
for (Module::const_iterator i = m.get()->getFunctionList().begin(),
     e = m.get()->getFunctionList().end(); i != e; ++i)
{
    if (!i->isDeclaration())
    {
        outs() << "Function name: " << i->getName() << "\n";
        outs() << "Number of lines: " << i->getInstructionCount() << "\n";
    }
}

return 0;
```

*Figure 1: Functionality code snippet*

The output is captured as following:

```
llvm@llvm:~/ECET480/tutorial/llvm_inter_tools$ cd ..
llvm@llvm:~/ECET480/tutorial$ cd first_llvm_project/
llvm@llvm:~/ECET480/tutorial/first_llvm_project$ ls
hello  hello.cc  makefile  sum.linked.bc
llvm@llvm:~/ECET480/tutorial/first_llvm_project$ ./hello sum.linked.bc
Function name: main
Number of lines: 8
Function name: sum
Number of lines: 8
llvm@llvm:~/ECET480/tutorial/first_llvm_project$
```

*Figure 2: Program output*

After iterating through the function list, the two functions were “main” and “sum” and the number of lines of instructions in each function were both 8.

## II. Verify the correctness of the output:

By inspecting the two files: main.ll and sum.ll, it can be easily inspected that each function has 8 lines of instructions inside its function definition.

```
; ModuleID = 'main.bc'
source_filename = 'main.c'
target datalayout = "e-m:e-p270:32:32-p271:32:32-p272:64:64-i64:64-f80:128-n8:16:32:64-S128"
target triple = "x86_64-unknown-linux-gnu"

@.str = private unnamed_addr constant [4 x i8] c"%d\\0A\\00", align 1

; Function Attrs: noinline nounwind optnone uwtable
define dso_local i32 @main() #0 {
    %1 = alloca i32, align 4
    %2 = alloca i32, align 4
    store i32 0, i32* %1, align 4
    %3 = call i32 @sum(i32 3, i32 4)
    store i32 %3, i32* %2, align 4
    %4 = load i32, i32* %2, align 4
    %5 = call i32 (i8*, ...) @printf(i8* getelementptr inbounds ([4 x i8], [4 x i8]* @.str, i64 0, i64 0), i32 %4)
    ret i32 0
}

declare dso_local i32 @sum(i32, i32) #1

declare dso_local i32 @printf(i8*, ...) #1

attributes #0 = { noinline nounwind optnone uwtable "frame-pointer"="all" "no-trapping-math"="true" "stack-protector-buffer-size"="8" "target-cpu"="x86-64" "target-features"="+cx8,+fxsr,+mmx,+sse,+sse2,+x87" "tune-cpu"="generic" }
attributes #1 = { "frame-pointer"="all" "no-trapping-math"="true" "stack-protector-buffer-size"="8" "target-cpu"="x86-64" "target-features"="+cx8,+fxsr,+mmx,+sse,+sse2,+x87" "tune-cpu"="generic" }

!llvm.module.flags = !{!0}
!llvm.ident = !{!1}

!0 = !{i32 1, !"wchar_size", i32 4}
!1 = !{!"clang version 13.0.0 (https://github.com/llvm/llvm-project.git d6a0560bf258f95f8960f35657a454f26dda5ba3)"}

"main.ll" 31L, 1372C                                1,1      All
```

Figure 3: Function main() code inspection

```
; ModuleID = 'sum.bc'
source_filename = 'sum.c'
target datalayout = "e-m:e-p270:32:32-p271:32:32-p272:64:64-i64:64-f80:128-n8:16:32:64-S128"
target triple = "x86_64-unknown-linux-gnu"

; Function Attrs: noinline nounwind optnone uwtable
define dso_local i32 @sum(i32 %0, i32 %1) #0 {
    %3 = alloca i32, align 4
    %4 = alloca i32, align 4
    store i32 %0, i32* %3, align 4
    store i32 %1, i32* %4, align 4
    %5 = load i32, i32* %3, align 4
    %6 = load i32, i32* %4, align 4
    %7 = add nsw i32 %5, %6
    ret i32 %7
}

attributes #0 = { noinline nounwind optnone uwtable "frame-pointer"="all" "no-trapping-math"="true" "stack-protector-buffer-size"="8" "target-cpu"="x86-64" "target-features"="+cx8,+fxsr,+mmx,+sse,+sse2,+x87" "tune-cpu"="generic" }

!llvm.module.flags = !{!0}
!llvm.ident = !{!1}

!0 = !{i32 1, !"wchar_size", i32 4}
!1 = !{!"clang version 13.0.0 (https://github.com/llvm/llvm-project.git d6a0560bf258f95f8960f35657a454f26dda5ba3)"}

"sum.ll" 24L, 946C                                8,1      All
```

Figure 4: Function sum() code inspection

Therefore, the program output was correct with both functions' name and lines of instructions.

