

TDT4205 Problem Set 5

Spring 2017

Answers are to be submitted through *It's Learning*, by March 28th, 20:00.

1 x86_64 assembly

The program below (inefficiently) finds prime factors of 1836311903. Write equivalent x86_64 assembler code.

```
#include <stdio.h>
#include <stdint.h>

void
factor ( int64_t n )
{
    int64_t f;
    for ( f=n>>1; ( n%f )!=0; f--);
    if ( f!=1 )
        factor ( f ), factor ( n/f );
    else
        printf ( "%ld is a prime factor\n", n );
}

int
main ()
{
    factor ( 1836311903 );
}
```

2 VSL code generation

The VSL compiler in the provided archive ps5_skeleton.tgz is extended with a function 'generate_program' in generator.c; this function is called from main.c, after syntax tree and symbol table construction. Implement this function so that it generates x86_64 assembly code for the following constructs:

2.1 Global string table

Strings should be given numbered labels in a data segment.

2.2 Global variables

Global variables should be given names corresponding to their declarations, prefixed with an underscore character '_', so as to avoid names that clash with names from the system libraries.

2.3 Functions

Functions should be placed in the text segment, named in the same manner as global variables, and set up/remove a stack frame. Furthermore, they should initiate a recursive traversal of their syntax subtrees, so that the remaining constructs can be generated.

2.4 Function parameters

Function parameters should be expected to follow the standard calling convention covered in lectures. Copies can be placed at the bottom of the function's stack frame, to make their run-time address computable from their sequence number, and liberate the registers for further function calls.

2.5 Arithmetic expressions

Arithmetic expressions should be translated so as to leave their result in the RAX register, and remove any intermediate calculations from the generated program's run time stack.

2.6 Assignment statements

Assignment statements should copy the result of an expression to the address of the assigned variable.

2.7 PRINT statements

PRINT statements can be translated into a sequence of 'printf' calls, with one call per item in the PRINT statement's list.

2.8 RETURN statements

RETURN statements should leave the result of their expression in the RAX register, remove the function's stack frame, and return control to the caller.

The vsl_programs directory contains one program which uses only this subset of the source language, and should result in a working binary when compiled/assembled. Two auxiliary routines are supplied in generator.c to assist with generating the code which starts the program when it is called from an external shell: consult the PS5 recitation slides on *it's learning* for suggestions on implementation.