Marmara University – Faculty of Engineering

Computer Engineering

Fall 2020-2021

CSE 2138 – Systems Programming

Midterm Exam

11 December 2020

HONOR CODE

On my honor, I have neither given nor received any unauthorized and/or inappropriate assistance for this exam. The work done on this exam is totally my own. I understand that by the school code, violation of these principles will lead to a zero grade and is subject to harsh discipline issues.

Notes on Solving the Exam Questions and Submitting:

1. Book, slides and notes are open. **Calculator or converter usage is not allowed.**
2. Internet usage is not allowed; however, computers and cell phones may be used only for viewing course materials provided for this course.
3. This exam consists of 5 questions.
4. Please write your answers on A4 size white blank sheets, with a **legible handwriting**.
5. **Show all your work to get full/partial points. Only the final answers will not be graded with full points!**
6. At the top of your first answer sheet, you must write your student ID, name, surname, and the "honor code" (exactly) and sign.
7. Scan all the solution pages to a single PDF file, name it as "FirstName_LastName_ID.pdf".
8. Upload your PDF file to http://ues.marmara.edu.tr before 11:00 o'clock.

1. **(18 pts)** Assume that:
   - We are running code on a machine using 12-bit two's complement arithmetic for signed integers
   - `short` integers are encoded using 6 bits

   The following definitions are given:

```
int x = -307;
unsigned u = x;
short s = (short)-101;
short sx = (short)x;
```

   Fill in the empty boxes in the tables below.

   Hint: Be careful with the rules that C uses for signed and unsigned ints.

   Show all your work!

	Expression	Decimal Representation (i)	Binary Representation (ii)
	`0`	0	0000 0000 0000
a)	`x`	-307	
b)	`u`		
c)	`s`		
d)	`u / 2`		
e)	`s >> 1`		
f)	`sx`		
g)	`(!u - 1) & u`		
h)	`((int)(u >> 11) + ~0)`		
i)	`((int)u) >> 11`		

Note: Write your final answers for corresponding empty cells like:

a.i) a.ii) b.i) …

2. **(24 pts)** Consider two different 9-bit floating point formats based on the IEEE floating point format given below:

   Format A:
   - There is sign bit in the most significant bit.
   - The next 3 bits are the exponent. The bias is $2^{k-1} - 1 = 3$.
   - The last 5 bits are the fraction.

   Format B:
   - There is sign bit in the most significant bit.
   - The next 4 bits are the exponent. The bias is $2^{k-1} - 1 = 7$.
   - The last 4 bits are the fraction.

   For formats A and B, fill in the empty boxes in the table below (use round-to-even when necessary). You can express numerical values as fractions (e.g. 183/256).

   Show all your work!

		ii)	ii)
	Value	**Format A Bits**	**Format B Bits**
	Zero	0 000 00000	0 0000 0000
a)	6		
b)		1 010 11001	
c)			0 1010 1001
d)	8.75		

Note: Write your final answers for corresponding empty cells like:

a.i)

a.ii)

b.i)

…

3. **(30 pts)** Consider the following C code and the assembly code that implements the branches of the switch statement of C code:

```
long fun(long x, long y){
   long result = 0;
   switch (x) {
        case ________:
                a)
              result = ________;
                          b)
              break;
        case ________:
                c)
              y = _________;
                    d)
        case ________:
                e)
              result = ________;
                          f)
              break;
        case ________:
                g)
        case ________:
                h)
              result = ________;
                          i)
              break;
        default:
              result = ________;
                          j)
   }
   return result;
}
```

```
f60:     movq  %rsi, %rax
f63:     cmpq  $6, %rdi
f67:     ja    0xf73
f69:     jmpq *0x5a0(,%rdi,8)
f6a:     addq  %rdi, %rax
f6d:     retq
f6e:     shlq $2, %rax
f72:     retq
f73:     leaq (%rax,%rax,2), %rax
f77:     retq
f78:     orq   %rdi, %rax
f7b:     incq %rax
f8e:     retq
```

The jump table for the switch statement is given below.
You can assume that the first entry in the jump table is for the case when y=0.
Parameters `x` and `y` are passed in registers `%rdi`, and `%rsi`, respectively.

```
0x5a0:  0xf6a
0x5a8:  0xf6e
0x5b0:  0xf73
0x5b8:  0xf73
0x5c0:  0xf78
0x5c8:  0xf6a
0x5d0:  0xf7b
```

Fill in the blank portions of the C code above to reproduce the function corresponding to the assembly code.

Show all your work!

4. **(18 pts)** Consider the following C function and its x86-64 assembly code. Fill in the missing parts of the C code to get a program equivalent to the given assembly code. **Show all your work!**

```
long foo(long m, long n, long k){

  long i=___________;
            a)

  while (___________){
            b)

        if(_____________){
              c)

             m = _____________;
                      d)
        }

        __________;
            e)

  }

  return __________;
            f)

}
```

```
    pushq %rbp
    movq  %rsp, %rbp
    movq  %rdi, %rax
    testq %rsi, %rsi
    jle   LBB0_3
    movq  %rsi, %rcx
LBB0_2:
    xorl  %edi, %edi
    cmpq  %rdx, %rax
    setg  %dil
    subq  %rdi, %rax
    decq  %rcx
    jne   LBB0_2
LBB0_3:
    addq  %rsi, %rax
    popq  %rbp
    retq
```

5. **(10 pts)** Consider the following C code and the assembly code for the same function:

```
_rec:
  pushq  %rbp
  movq   %rsp, %rbp
  movl   $1, %eax
  cmpq   $rsi, %rdi
  je     LBB0_3
  movq   %rsi, %rcx
  subq   %rdi, %rcx
  movl   $1, %eax
LBB0_2:
  incq   %rax
  addq   %rsi, %rcx
  jne    LBB0_2
LBB0_3:
  popq   %rbp
  retq
```

```
long rec(long x, long y){

  if (_______________)
            a)

    return __________;
              b)
  else{

    long a;
    a = rec(_______, _______);
              c)        d)

    return ____________;
              e)
  }

}
```

Fill in the missing parts of the C code to get a program equivalent to the generated assembly code. **Show all your work!**