Full Name: Zachary Wilcox

A-number: A02080016

ECE 5720, Fall 2020

Take Home 1

Due: September 15, 2020 (3:00 PM)

Instructions:

- Write your A-number on top of every sheet.
- Make sure that your exam is not missing any sheets, then write your full name on the front.
- The exam has a maximum score of 20 points. You must show your steps clearly to get any credit. Good luck!

1 (10):
2 (10):
TOTAL (20):

Problem 1. (10 points):

Condider the following assembly code for a C `for` loop:

🟥 compare quad

🟨 ≥ >=

🟧 h

🟪 len

🟩 t

🟩 t's value

🟪 clear

```
loop:
        movslq      %esi, %rsi
        leaq       -1(%rdi,%rsi), %rax
        cmpq       %rax, %rdi
        jnb        .L1
.L5:
        movzbl      (%rax), %edx     t
        subq       $1, %rax     t--
        addb       (%rdi), %dl   ← t+h
        movb       %dl,  (%rdi)
        xorb       1(%rax), %dl   ← t^(t+h)
        movb       %dl, 1(%rax)
        xorb       %dl,  (%rdi)  h ^= t
        addq       $1, %rdi    h++
        cmpq       %rax, %rdi  compare
        jb        .L5    Jump
.L1:
        rep ret
```

Copy t to edx

Sub 1 from t

add h to dl

Based on the assembly code above, fill in the blanks below in its corresponding C source code. (Note: you may only use the symbolic variables `h`, `t` and `len` in your expressions below — *do not use register names.*)

```
void loop(char *h, int len)
{
   char *t;

   for (t=(h+len-1); t<h ; h++,t--) {
      h += t ;
      t ^= (t+h);
      h ^= t ;
   }

   return;
}
```

h = dl

(t+1)^dl

move dl into (t+1)

dl ^ h

h+1

Compare t&h

if t<h, repeat

🟨 x

🟧 y

🟪 result

Compare x&y

<=

Problem 2. (10 points):

Condider the following assembly code for a C `for` loop:

```
decode_me:
        cmpl        %esi, %edi     compare y,x
                                     <=
        jle        .L4
        movl        %edi, %edx     x-val
        movl        $1, %eax       result=1
        subl        %esi, %edx     (x-y)
.L3:
        subl        $1, %edi
        addl        $1, %esi
        imull        %edx, %eax     result *=(x-y)
        subl        $2, %edx        update x-y
        cmpl        %esi, %edi      compare y,x
        jg         .L3
        rep ret
.L4:
        movl        $1, %eax
        ret
```

Based on the assembly code above, fill in the blanks below in its corresponding C source code. (Note: you may only use the symbolic variables `x`, `y`, and `result` in your expressions below — *do not use register names.*)

```
int decode_me(int x, int y)
{
   int result;

   for (result = 1; X > y; x--,y++ ) {

     result *=(x-y);

   }


   return result;
}
```