Machine-Level Representation of Programs #4 4190.308 Computer Architecture

Name: _____

Due Date: Thursday, October 9, 2014, 23:59 **Student-Number:** _____

Submission: in paper form.
There will be a drop off box in class and in front of the CSAP Lab in building 301, room 419.

Question 1

x86-64 Procedures

For the C functions *fun1* and *fun2* with the following general structure GCC generates the x86-64 assembly code as shown below:

```
void fun1(long *arr, long size, long val)
{
  long i;
  for(i = 0; i < size; i++) {
    arr[i] += val;
  }
}

void fun2()
{
  long arr[3] = {1, 3, 5};
  long size = 3;
  long val = 5;
  fun1(arr, size, val);
}
```

```
fun1:
  pushq %rbp
  movq  %rsp, %rbp
  movq  %rdi, -24(%rbp)
  movq  %rsi, -32(%rbp)
  movq  %rdx, -40(%rbp)
  movq  $0, -8(%rbp)
  jmp .L2
.L3:
  movq  -8(%rbp), %rax
  salq  $3, %rax
  addq  -24(%rbp), %rax
  movq  -8(%rbp), %rdx
  salq  $3, %rdx
  addq  _____(%rbp), %rdx
  movq  (%rdx), %rdx
  addq  _____(%rbp), %rdx
  movq  %rdx, (%rax)
  addq  $1, -8(%rbp)
.L2:
  movq  -8(%rbp), %rax
  cmpq  _____(%rbp), %rax
  jl  .L3
  leave
  ret
fun2:
  pushq %rbp
  movq  %rsp, %rbp
  subq  $48, %rsp
  movq  $1, -48(%rbp)
  movq  $3, -40(%rbp)
  movq  $5, -32(%rbp)
  movq  $3, -8(%rbp)
  movq  $5, -16(%rbp)
  movq  -16(%rbp), ________
  movq  -8(%rbp), %rcx
  leaq  -48(%rbp), %rax
  movq  %rcx, _________
  movq  %rax, _________
  call  fun1
  leave
  ret
```

Fill in the missing part in the x86-64 assembly code shown above.

Question 2
wrap up

Write a “fun” function as shown in given skeleton.

You can implement the function to do anything you want, but meet the following conditions.

1. include at least one branch statement

2. include at least one nested loop

3. the assembly code of output file should include at least one conditional move instruction, ‘`CMOVcc`’.

< Skeleton Code of C >

```
int fun(int from, int to)
{
  int result;

  // TODO: Implement this function

  return result;
}

int main(int argc, char* argv[])
{
  int result = fun(1, 10);
  return 0;
}
```

<Compile Options>

The codes you have submitted are to be compiled by the following command.

```
gcc -S -m32 -O0 main.c
```

Then, the output file, “main.s”, will be genearted.

Make a table and fill the table C code at left hand side and the assembly code at right hand side. Match the x86-64 assembly code to C code and explain like an example.

< Example >

```
int fun(unsigned x) {
  int val = 0;
  int i;
  for (i = 1; i < x+1; i++) {

    if(i%2 == 0) {

      val += i;
    }
  }
  return val;
}
```

For statement

Procedure return

```
x at %ebp+8

fun:
  pushl %ebp
  xorl  %eax, %eax
  movl  %esp, %ebp
  movl  $1, %edx
  pushl %ebx
  movl  8(%ebp), %ebx
  addl  $1, %ebx
  cmpl  $1, %ebx
  ja  .L7
  jmp .L2
.L5:
  leal  (%eax,%edx), %ecx
  testb $1, %dl
  cmove %ecx, %eax
.L7:
  addl  $1, %edx
  cmpl  %edx, %ebx
  ja  .L5
.L2:
  popl  %ebx
  popl  %ebp
  ret
```