Homework 3

Huy Quang Lai
132000359

Texas A&M University

1 November 2022

An Aggie does not lie, cheat or steal.
Nor does an Aggie tolerate those who do.

Chapter 3

3.5

Correct

```
void decode1(long* xp, long* yp, long* zp) {
    long x = *xp;
    long y = *yp;
    long z = *zp;

    *yp = x;
    *zp = y;
    *xp = z;
}
```

3.6

Correct

Instruction	Result
`leaq 9(%rdx), %rax`	$9 + q$
`leaq (%rdx,%rbx), %rax`	$q + p$
`leaq (%rdx,%rbx,3), %rax`	$q + 3p$
`leaq 2(%rbx,%rbx,7), %rax`	$2 + 8p$
`leaq 0xE(,%rdx,3), %rax`	$14 + 3p$
`leaq 6(%rbx,%rdx,7), %rdx`	$6 + p + 7q$

3.7

Correct

```
leaq (%rsi,%rsi,9), %rbx     \\ %rbx = 10 * y
leaq (%rbx,%rdx), %rbx       \\ %rbx += %rdx
leaq (%rbx,%rdi,%rsi), %rbx \\ %rbx += y * x
```

```
short scale3(short x, short y, short z) {
    short t =  10 * y + z + y * x;
    return t;
}
```

3.8

Correct

Instruction	Destination	Value
addq %rcx,(%rax)	0x100	0x100
subq %rdx,8(%rax)	0x108	0xA8
imulq $16,(%rax,%rdx,8)	0x118	0x110
incq 16(%rax)	0x110	0x14
decq %rcx	%rcx	0x0
subq %rdx,%rax	%rax	0xFD

3.18

Correct

```
short test(short x, short y, short z) {
    short val = z + y - x;
    if (z > 5) {
        if (y > 2)
            val = x/z;
        else
            val = x/y;
    } else if (z < 3)
        val = z / y;
    return val;
}
```

3.20

Correct
The operator is division '/'

```
arith:
    leaq    15(%rdi), %rbx  // temp = x + 15
    testq   %rdi, %rdi      // test x
    cmovns  %rdi, %rbx      // if x >= 0, temp = x
    sarq    $4, %rbx        // result = temp >> 4 (= x / 16)
    ret                     // return
```

3.24

Correct

```
cmpq %rsi, %rdi => %rdi (a) > %rsi (b)
leaq (,%rsi,%rdi), %rdx => %rdx + 0 + %rsi * %rdi
```

```
short loop_while(short a, short b) {
    short result = 0;
    while (a > b) {
        result = result + (a * b);
        a = a - 1;
    }
    return result;
}
```

3.25

Correct

```
long long_while2(long a, long b) {
    long result = b;
    while (b > 0) {
        result = result * a;
        b = b - a;
    }

    return result
}
```

3.32

Correct

Instruction			States values (at beginning)					
Label	PC	Instruction	%rdi	%rsi	%rax	%rsp	*%rsp	Description
M1	0×400560	callq	10	-	-	0×7fffffffe820	-	Call first(10)
F1	0×400548	lea	10	-	-	0×7fffffffe818	0×400565	Entry of first
F2	0×40054c	sub	10	11	-	0×7fffffffe818	0×400565	
F3	0×400550	callq	9	11	-	0×7fffffffe818	0×400565	Call last(9, 11)
L1	0×400540	mov	9	11	-	0×7fffffffe810	0×400555	Entry of last
L2	0×400543	imul	9	11	9	0×7fffffffe810	0×400555	
L3	0×400547	retq	9	11	99	0×7fffffffe810	0×400555	Return 99 from last
F4	0×400555	repz repq	9	11	99	0×7fffffffe818	0×400565	Return 99 from first
M2	0×400565	mov	9	11	99	0×7fffffffe820	-	Resume main

Figure 1: 3.32

3.35

Correct

Register `%rbx` holds the value of parameter `x`. `%rbx` will be used to compute the result expression.

```
long rfun(unsigned long x) {
    if (x == 0)
        return 0;
    unsinged long nx = x >> 2;
    long rv = rfun(nx);
    return x + rv;
}
```

3.37

Correct

Expression	Type	Value	Assembly Code
`P[1]`	short	$M[x_p + 2]$	`movw 2(%rdx),%ax`
`P3+i+`	short *	$x_p + 6 + 2i$	`leaq 6(%rdx,%rcx,2),%rax`
`P[i*6-5]`	short	$M[x_p + 12i - 10]$	`movw -10(%rdx,%rcx,12),%rax`
`P[2]`	short	$M[x_p + 4]$	`movw 4(%rdx),%ax}`
`&P[i+2]`	short *	$x_p + 2i + 4$	`leaq 4(%rdx,%rcx,2),%rax`

3.38

Correct

```
sum_element:
    leaq 0(,%rdi,8), %rdx       \\ Compute 8i
    subq %rdi, %rdx             \\ Compute 7i
    addq %rsi, %rdx             \\ Compute 7i + j
    leaq (%rsi,%rsi,4), %rax    \\ Compute 5j
    addq %rax, %rdi             \\ Compute i + 5j
    movq Q(,%rdi,8), %rax       \\ Retrieve M[x_q + 8(5j+ i)]
    addq P(,%rdx,8), %rax       \\ Add M[x_p + 8(7i + j)]
```

`P` has a byte offset of $8 \cdot (7i + j)$
`Q` has a byte offset of $8 \cdot (5j + i)$

3.41

Correct
`p` → [0, 8]
`s.x` → [8,10]
`s.y` → [10,12]
`next` → [12,20]

The structure requires a total of 20 bytes

```
void st_init(struct test *st) {
    st->s.y  = st->s.x;
    st->p    = &(st->s.y);
    st->next = st;
}
```