시스템프로그램 중간시험 (2021학년도 2학기, 10/21/2021)

★ **답안지의 각 페이지 상단에** 본인 소속 **학과명, 학번, 이름**을 기재하세요.
★ 답안지에 문제를 작성할 필요가 없으며, 각 문항 번호에 맞춰 답만 작성합니다. (필요한 경우 표로 답안 작성 가능)

(1) Assume we are running code on a 6-bit machine using 2's-complement arithmetic for signed integers. A short integer is encoded using 3-bits. Fill in the empty boxes in the table below. You need not fill in entries marked "—". The following definitions are used in the table:

```
short sy = -4;
int y = sy;
int x = -16;
unsigned ux = x;
```

Expression	Decimal value	Binary representation
Zero	0	000000
—	10	001010
—	-18	⑥
ux	①	⑦
y + 3	②	⑧
x >> 3	③	⑨
-TMax	④	—
-TMin	—	⑩
TMax + TMin + 1	⑤	—

(2) Consider the following 16-bit representation based on IEEE floating point format.
- There is a sign bit in the MSB
- The next 7 bits are the exponent
- The last 8 bits are the significand

The rules are like those in the IEEE standard (normalized, denormalized, zero, infinity, and NaN), where we consider the floating point format to encode numbers in a form

$$(-1)^s \times m \times 2^E$$

where m is the mantissa and E is the exponent.
Fill in the table below for the descriptions given, with the following instructions for each column.
- Hex: 4 hexadecimal digits
- m: The fractional value of the mantissa in the form of x/y, where x is an integer and y is an integral power of 2
- E: The integer value of the exponent
- Value: The numeric value represented, in the form of x * 2^z, where x and z are integers

Description	Hex	m	E	Value
-0	ⓐ	ⓕ	ⓚ	—
Smallest value > 1	ⓑ	ⓖ	ⓛ	ⓟ
Largest denormalized	ⓒ	ⓗ	ⓜ	ⓠ
Largest normalized	ⓓ	ⓘ	ⓝ	ⓡ
+∞	ⓔ	—	—	—
Number 0x3AA0	—	ⓙ	ⓞ	ⓢ

(3) In the following questions, assume the variables **a** and **b** are signed integers and that the machine uses 2's complement representation. Also assume **MAX_INT** is the maximum integer, **MIN_INT** is the minimum integer, and **W** is one less than the word length (eg, **W**=31 for 32-bit integers). Match each of the descriptions on the left with a line of code on the right (write in letter).

Description	Answer	Code
① 1's complement of a		a. `a ^ (MIN_INT + MAX_INT)`
② a		b. `~((a >> W) << 1)`
③ a & b		c. `1 + (a << 3) + ~a`
④ a * 7		d. `(a << 4) + (a << 2) + (a << 1)`
⑤ a / 4		e. `((a < 0) ? (a + 3) : a) >> 2`
⑥ (a < 0) ? 1 : -1		f. `~(~a | (b ^ (MIN_INT + MAX_INT)))`
		g. `~((a | (~a + 1)) >> W) & 1`
		h. `((a ^ b) & ~b) | (~(a ^ b) & b)`
		i. `a >> 2`

(4) For a function `decode4()`, **gcc** generates the following assembly code in the right side. Parameters x, y, and z are passed in registers %rdi, %rsi, and %rdx. The code stores the return value in register %rax. Fill in the C code for `decode4` that will have an effect equivalent to the assembly code shown.

```
long decode4(long x, long y, long z)
{
   long t1 = y - z;
   long t2 = ___________;      // ①
   long t3 = (t1 << 63) >> 63;
   long t4 = ___________;      // ②
   return t4;
}
```

```
decode4:
   subq %rdx, %rsi
   imulq %rsi, %rdi
   movq %rsi, %rax
   salq $63, %rax
   sarq $63, %rax
   xorq %rdi, %rax
   ret
```

①

②

(5) For C code having the form, shown on the left side of the following table, **gcc**, run with the command-line option **–O1**, produces the code, shown on the right side of the table. Fill in the missing parts of the C code. (Note that the control structure in the assembly code does not exactly match what would be obtained by a direct translation of the C code according to the guarded-do translation rules. However, you can fill out the missing parts of the C code by understanding the relationships of the codes.)

```
long loop5(long a, long b)
{
   long res = __________;    // ①
   while (__________) {      // ②
      res = __________;      // ③
      b = __________;        // ④
   }
   return res;
}
```

```
loop5:
   testq %rsi,%rsi
   jle .L8
   movq %rsi,%rax
.L7:
   imulq %rdi,%rax
   subq %rdi,%rsi
   testq %rsi,%rsi
   jg .L7
   rep; ret
.L8:
   movq %rsi,%rax
   ret
```

(6) In the following C code on the left-side, A and B are constants defined with #define, and **gcc** generates the following right-upper-side code for the function **setVal**. In this case, what are the possible values of A and B?

```
typedef struct {
   int x[A][B]; /* Unknown constants A and B */
   long y;
} str1;

typedef struct {
   char array [B];
   int t;
   short s[A];
   long u;
} str2;

void setVal(str1 *p, str2 *q) {
   long v1 = q->t;
   long v2 = q->u;
   p->y = v1 + v2;
}
```

```
void setVal(str1 *p, str2 *q)
p in %rdi, q in %rsi
setVal:
   movslq 8(%rsi), %rax
   addq 32(%rsi), %rax
   movq %rax, 224(%rdi)
   ret
```

☆ 수고했습니다~.