LAB02-B

Q1

The following assembly code:

Initially x, y, and n are offsets 8, 12, and 16 from %ebp

```
movl 8(%ebp),%ebx
movl 16(%ebp),%edx
xorl %eax,%eax 4
decl %edx 5
__________ (jump instruction)
movl %ebx,%ecx
imull 12(%ebp),%ecx
.p2align 4,,7  Inserted to optimize cache performance
.L6:
addl %ecx,%eax
subl %ebx,%edx
_______ (jump instruction)
.L4:
```

was generated by compiling C code that had the following overall form

```
int loop(int x, int y, int n) {
 int result = 0;
 int i;
 for (i = ____; i >= 0 ; i = ___ ) {
     result += _____ ;
}
 return result;
}
```

Your task is to fill in the missing parts of the C code to get a program equivalent to the generated assembly code.

Q2

For a function with prototype

```
int decode2(int x, int y, int z);
```

GCC generates the following assembly code:

```
decode2:
  subl %edx,%esi
  imull %esi,%edi
  movl %esi,%eax
  sall $31,%eax
  sarl $31,%eax
  xorl %edi,%eax
  ret
```

Parameters x, y, and z are passed in registers `%edi`, `%esi`, and `%edx`. The code stores the return value in register `%eax`.

Write C code for `decode2` that will have an effect equivalent to the assembly code shown.

Q3

Suppose we number the bytes in a `w-bit` word from `0` (least significant) to `w/8 - 1` (most significant). Write code for the following C function, which will return an unsigned value in which byte `i` of argument `x` has been replaced by byte `b`:

```
unsigned replace_byte (unsigned x, int i, unsigned char b);
```

Example:

- replace_byte(0x12345678, 0, 0xAB) --> 0x123456AB
- replace_byte(0x12345678, 3, 0xAB) --> 0xAB345678

Q4

Suppose we are given the task of generating code to multiply integer variable `x` by various different constant factors `K`. To be efficient, we want to use only the operations `+`, `-`, and `<<`. For the following values of `K`, write C expressions to perform the multiplication using at most three operations per expression.

A. K = 31

B. K = –3

C. K = 1032

D. K = –48

Q5

Following the bit-level floating-point coding rules, implement the function with the following prototype:

```
typedef unsigned float_bits;
/* Compute |f|. If f is NaN, then return f. */
float_bits float_absval(float_bits f);
```

For floating-point number `f` , this function computes `|f|` . If f is `NaN` , your function should simply return `f` .

Q6

- 阅读《汇编语言》（王爽著）第一、二章。
- 在本机下载安装dosbox，配置好实验环境，为下次上机实验做准备。