1. Data Representation

A. Give the binary representation of these numbers in 4-bit, signed 2's complement.

- 5
- –1
- –3
- –8

B. Perform these additions in 4-bit binary. After each addition, note whether any of these conditions hold:

ZF — if the result is zero
SF — if the result is negative (when interpreted as 2's complement)
CF — if the result is too large (when interpreted as unsigned binary)
OF — if the result is too large (when interpreted as 2's complement)

- (Example) 1111 + 0001 = **0000 ZF CF**
- 1010 + 0010 =
- 0111 + 0001 =
- 1110 + 1110 =
- 0010 + 1111 =

C. Given a 7-bit floating point representation comprising 1 sign bit, 3 exponent bits, and 3 significand bits, interpret the following values by giving *m* (in binary) and e (in decimal) such that the number equals m × 2e, or write ±∞ or NaN.

- Example: 0 110 000 = **1.000×2^3**
- 0 001 110
- 1 111 000
- 1 010 000
- 0 000 110

2. IA32 Assembly Language

Consider the following fragment of assembly code:

```
pushl   %ebp
movl    %esp, %ebp
movl    0x8(%ebp), %ecx
movl    0xC(%ebp), %edx
movl    (%ecx), %eax
addl    %edx, (%ecx)
popl    %ebp
ret
```

Assuming initial values of the registers and of memory as shown in the table below (all values in hexadecimal). As the code executes, show the changed value for each memory location or register by writing the new value to its right. Some values may not change, and others may change more than once.

Exception: for %eip, only give the final value after the `ret`.

%eip	
%esp	FFFF1004
%ebp	FFFF1018
%eax	00000066
%ebx	000F8000
%ecx	FFFFFFFF
%edx	FFFFFFED
FFFF101C	FFFFE770
FFFF1018	FFFF1030
FFFF1014	00000017
FFFF1010	00001000
FFFF100C	00000010
FFFF1008	FFFF1010
FFFF1004	FFFFFACE

3. C Programming

Write a function `multvm` that multiplies a matrix by a vector. The inputs are the dimensions of the matrix, the matrix X, and the vectors Y and Z. Set $Z = XY$. Assume that X, Y, and Z have already been allocated and that the number of rows and columns given are accurate.

The formula for multiplying a matrix by a vector is $Z_r = \sum_c (X_{rc} Y_c)$.

```
void multvm(int rows, int cols, double **X, double *Y, double *Z) {
```

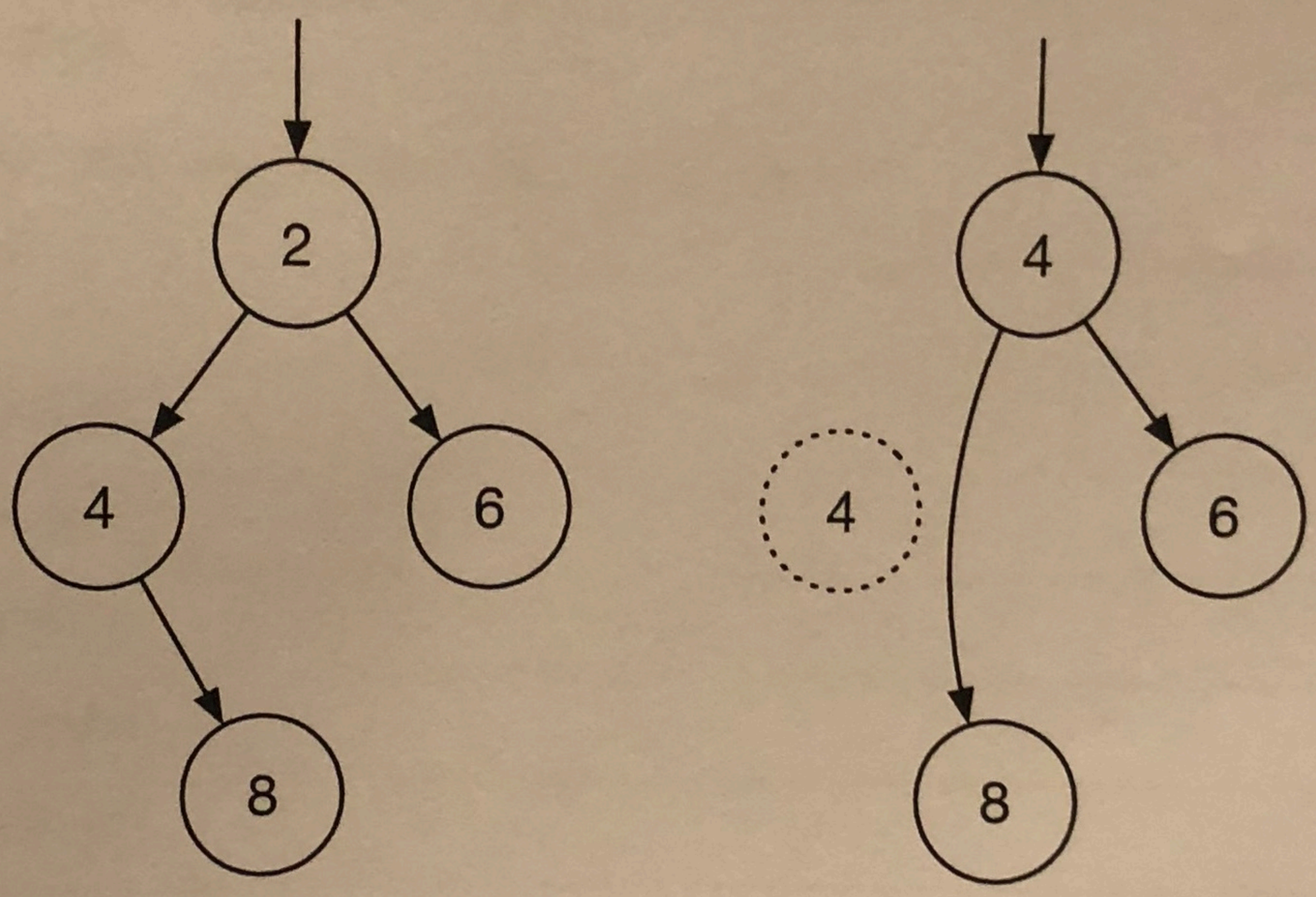
4. C Programming

Write a function `deletemin` which extracts the minimum element from a priority queue. The queue is represented as a binary tree with the heap property, meaning that the value of each node is less than the values of its children (if any). If *q* is null or the heap is empty, deletemin should return 1 and leave *min* unchanged. If the heap is not empty, deletemin should return 0, put the minimum element in *min*, and shrink the root.

To shrink a node with two children, set the key of the node equal to the smaller of its children's keys, and then shrink that child.

To shrink a node with less than two children, remove it from the tree and free it.

The two diagrams below show the structure of the queue before and after calling deletemin. After the call, min points to 2. The value returned is 0.



Assume these definitions:

```
struct priorityQ {
    struct node *root;
}

struct node {
    int key;
    struct node *left;
    struct node *right;
}
```

Example usage:

```
int x;
if (deletemin(&x, priorities) == 0) {
    printf("Got %d\n", x);
}
```

```
int deletemin(int *min, struct priorityQ *q) {
```