Question 1.
/** Let x be the value currently in b[h]. Permute b[h..k] and return an int j that satisfies b[h..j-1] <= b[j] = x < b[j+1..k]. */
public static int partition(int[] b, int h, int k) {
    int j = h; int t = k;
    /* inv: b[h..j-1] <= b[j] = x <= b[t+1..k] */
    while (j < t) {
        if (b[j+1] <= b[j]) {
            Swap b[j+1] and b[j]; j = j+1;
        } else {
            Swap b[j+1] and b[t]; t = t-1;
        }
    }
    return j;
}

Question 2.
int p = h; int q = p;
while (q <= k) {
    if (b[q] % 2 != 0) {
        b[p] = b[q]; p = p + 1;
    }
    q = q + 1;
}

Question 3a. h __a1__

3b. /** see exam for spec */
public static void swap(int[][] b, int n, int h, int k, int p, int q) {
    for (int r = 0; r != n; r = r+1) {
        for (int c = 0; c != n; c = c+1) {
            // Swap b[h+r][k+c] with b[p+r][q+c]
            int temp = b[h+r][k+c];
            b[h+r][k+c] = b[p+r][q+c];
            b[p+r][q+c] = temp;
        }
    }
}

Question 4.
/** = a string containing n occurrences of char c.
   Precondition: n ≥ 0 */
public static String occ(int n, char c) {
    if (n == 0) return "";
    return c + occ(n-1, c);
}

/** Eg. the call p("2A0B3V") produces "AAVVV".
   Pre: s contains an even number of chars,
   and the first of each pair is a digit.
   Produce a String that, for each pair "ic" where 'i'
   is a digit, contains i occurrences of char c. */
public static String p(String s) {
    if (s.length() == 0) return "";
    return occ(s.charAt(0) - '0', s.charAt(1)) + p(s.substring(2));
}

Question 5. (a) super();
(b) 1. Create a folder of class Student; execute the constructor call Student("Doe", "Fall", 2006); and
yield as value of the expression the name of the new folder.
(c) s can be cast to Object, CornellPersonnel, and Student. Casting down to Student must be done explicitly, using (Student) s.
(d) The apparent class is CornellPersonnel; the real class is Student.
(e) v.get(i) instanceof Faculty
(f) this refers to the object (folder)—or rather its name-- in which it occurs.
(g) person’s name: a. Put in CornellPersonnel, with class Name.
b. Person’s address. In CornellPersonnel with class Address.
c. College they teach in: In Faculty, with class College.
d. Graduate degree program: In Grad, with class GraduateDegree.
e. Transcript: In Student, with class Transcript.
(h) A parameter is a variable that is declared in the header of a method. An argument is an expression
that appears within the parentheses of a method call.
(i) fi can be referenced anywhere within class Faculty and nowhere else.
(j) if (ob == null || !ob instanceof CornellPersonnel)
    return false;
CornellPersonnel cp= (CornellPersonnel) ob;
return cp.name.equals(this.name) &&
    cp.address.equals(this.address);

Question 6.
public class Faculty {
    
    /** If this faculty member is not lec's mentor
        make this faculty member lec's mentor. */
    public void addMentee(Lecturer lec) {
        if (m.contains(lec))
            return;
        m.add(lec);
        lec.makeMentor(this);
    }
    /** Make sure that this faculty member is
        not lec’s mentor – remove lec from this
        faculty member’s list if necessary. */
    public void removeMentee(Lecturer lec) {
        if (!m.contains(lec))
            return;
        m.remove(lec);
        lec.removeMentor();
    }
}

public class Lecturer {
    
    /** Make f be this Lecturer’s mentor
        (if f is already the mentor, there is nothing
        to do; if someone else is the mentor, first
        remove that mentor) */
    public void makeMentor(Faculty f) {
        if (mentor == f)
            return;
        if (mentor != null) {
            removeMentor();
        }
        mentor= f;
        f.addMentee(this);
    }
}