The following questions refer to the webpage shown in Appendix A.

Answer the following True / False questions (circle the correct answers): (10 points)

(i) The page can be implemented using a table with three columns [T / F]
(ii) More than one house can be selected at any given time [T / F]
(iii) The source code did not define a `<TITLE></TITLE>` [T / F]
(iv) When you click on Reset, any selected button(s) gets de-selected [T / F]
(v) A `<ul>` command was definitely used in this page [T / F]

The following questions refer to the code documented in Appendix B. The code calculates the radius of a circle in feet and inches, given the circle’s area (in square feet).

(a) Answer the following True / False questions (circle the correct answers): (10 points)

(vi) RoundObject is a superclass of Circle [T / F]
(vii) The one-parameter Circle constructor calls the one-parameter RoundObject constructor [T / F]
(viii) The method displayFeetAndInches requires a target instance [T / F]
(ix) The three different Circle constructors are examples of encapsulation [T / F]
(x) The Circle class must have a method that calculates the radius [T / F]
(b) Describe (in English sentences, as opposed to pseudocode) how displayFeetAndInches calculates the number of feet and inches to which the double precision floating point variable lgth is equivalent.

The Math.floor method returns the closest integer that is less than lgth. Hence, displayFeetAndInches takes the whole number part of lgth and assigns it to feet. The fraction part of lgth, given by lgth-feet, is scaled up by a multiple of 12 to give the number inches this corresponds to, and this number is rounded to the closest integer.

(c) Suppose there exists another subclass of RoundObject, called Sphere. The Sphere class extends the RoundObject class, and has methods comparable to that of the Circle class. Using UML notation for objects and their inter-relationships, make representations of the different classes and diagram the relationships between them.

```
RoundObject
radius: Double
RoundObject()
RoundObject(radius)
setRadius(radius)
getRadius()
displayFeetAndInches(lgth)

extends
*

Circle
area: Double
Circle()
Circle(area)
Circle(radius,area)
setArea(area)
calcRadius()
displayObject

extends
*

Sphere
volume: Double
Sphere()
Sphere(volume)
Sphere(radius, volume)
setVolume(volume)
calcRadius()
displayObject
```
(d) You are asked to describe the following methods in pseudo-code (try to keep this as close as possible to Java syntax).

(i) Describe an instance method in the Sphere class called calcRadius. The method should calculate the radius from the volume information and set the value of the radius variable in the Sphere instance. The volume of a sphere is given by the formula \((4/3)*\pi*(\text{radius}^3)\).

\[
\text{public void calcRadius ()} \{
\text{ \hspace{1cm} this.radius = Math.pow((0.75*volume)/Math.PI,(1.0/3.0));}
\}
\]

(ii) Modify displayFeetAndInches so that it is more sensitive to singular and plural values. For example, it should know when to use “feet” and when to use “foot”. HINT: Use conditional statements.

\[
\text{public static void displayFeetAndInches (double lgth) \{} \\
\quad \text{int feet = (int)Math.floor(lgth);} \\
\quad \text{int inches = (int)Math.round(12.0*(lgth - feet));} \\
\quad \text{System.out.print(feet);} \\
\quad \text{System.out.print((feet==1 ? " foot" : " feet");} \\
\quad \text{System.out.print(" and " + inches);} \\
\quad \text{System.out.println((inches==1 ? " inch." : " inches.");}}
\]

(iii) Describe a class method in the RoundObjects class called displayMilesFeetAndInches. The method should take as argument a double precision variable \(\text{lgth}\) that gives some distance in miles, then displays this input in miles, feet and inches. Note that there are 5280 feet to a mile.

\[
\text{public static void displayMilesFeetAndInches (double lgth) \{} \\
\quad \text{int miles = (int)Math.floor(lgth);} \\
\quad \text{System.out.print(miles);} \\
\quad \text{System.out.print((miles==1 ? " mile, " : " miles, ");} \\
\quad \text{System.out.print((miles==1 ? " mile, " : " miles, ");} \\
\quad \text{displayFeetAndInches(5280.0*(lgth-miles));}
\]
(e) Harry, Ron and Hermione come across an old dragon’s lair with numerous marble discs (with etchings indicating their respective top surface areas in square feet) and crystal spheres (with, what do you know(?!?), etchings indicating their respective volumes in cubic feet). They are interested in comparing any two objects and finding out which has the larger radius.

Suppose a `Sphere` class has been defined that not only extends the `RoundObject` class, but also implements the interface shown in Appendix C.

Describe, in pseudo code (again, as Java-like as possible), a `Demonstrate` class that would take two `RoundObject` instances (one Circle and one Sphere) and compare their radii. The comparison should tell you which has a bigger radius, and also give you the ratio of their radii. Feel free to use all methods in the `RoundObject`, Circle and Sphere classes.

```java
public static void main (String argv[]) {
    RoundObject disc = new Circle(100.0);
    disc.calcRadius();
    disc.displayObject();

    RoundObject sphe = new Sphere(300.0);
    sphe.calcRadius();
    sphe.displayObject();

    double ratio = disc.getRadius() / sphe.getRadius();

    System.out.println("The ratio of the disc's radius to the sphere's radius is " + ratio);

    if (ratio > 1) {
        System.out.println("Disc has larger radius");
    } else if (ratio < 1) {
        System.out.println("Sphere has larger radius");
    } else {
        System.out.println("The radii are equal");
    }
}
```
Suppose that you work for the company HPAnalysts (HP as in Harry Potter, not the usual suspect). You are interested in figuring out which two Hogwarts’ houses are most popular with the fans. But first, you wish to find out how many such choices there are in total, i.e., how many ways there are to pick two out of four houses.

(a) You know that there are a total of \( \binom{4}{2} \) (four choose two) ways to select two out of the four houses. And that \( \binom{n}{r} = \frac{n!}{(n-r)!r!} \), where \( n! = n(n-1)(n-2)\ldots(1) \). In pseudocode, write a class method called factorial that calculates \( n! \) using a loop. In another class method called choose, use the factorial method to get the value of \( \binom{n}{r} \).

(15 points)

```java
public static int factorial (int n) {
    int f = 1;
    for (int i=n; i>0; i--) f *= i;
    return f;
}

public static int choose (int n, int r) {
    return factorial(n)/(factorial(n-r)*factorial(r));
}
```

(b) (i) Explain why \( \binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r} \). What is the value of \( \binom{n}{r} \) when \( r=0 \)? What is its value when \( n=r \)?

(5 points)

The choosing of \( r \) objects from \( n \) can be subdivided into the choosing of \( r \) objects from \( (1+n) \). Examine the first object, it is either chosen or not. If it is chosen, we need only choose \( r-1 \) from the remaining \( n-1 \). If it is not chosen, we need to choose \( r \) from the remaining \( n-1 \). Hence, \( \binom{n}{r} = \binom{n-1}{r-1} + \binom{n-1}{r} \).

when \( r=0, \binom{n}{r}=1 \)
when \( r=n, \binom{n}{r}=1 \)

(ii) Now create a recursive version of the choose method, call it recursiveChoose. Again, describe your method using Java-like pseudocode.

(10 points)

```java
public static int recursiveChoose (int n, int r) {
    if (n==r) return 1;
    else if (r==0) return 1;
    else return recursiveChoose(n-1,r-1) + recursiveChoose(n-1,r);
}
```