Recommendation:

Play the game and attempt to answer the questions yourself without looking at the answers.

You’ll learn much less if you just look at the question, then the answer, and go “Okay, that makes sense” (as opposed to actively attempting to think about a solution first).
Choose a category.
You will be given the answer.
You must give the correct question.

Click to begin.
Choose a point value.
Choose a point value.

Click here for
Final Jeopardy
<table>
<thead>
<tr>
<th>Ominous OCaml</th>
<th>Data Structures</th>
<th>Typing &amp; Hierarchy</th>
<th>Jolly Java</th>
<th>Crafty Coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 Point</td>
<td>10 Point</td>
<td>10 Point</td>
<td>10 Point</td>
<td>10 Point</td>
</tr>
<tr>
<td>20 Points</td>
<td>20 Points</td>
<td>20 Points</td>
<td>20 Points</td>
<td>20 Points</td>
</tr>
<tr>
<td>30 Points</td>
<td>30 Points</td>
<td>30 Points</td>
<td>30 Points</td>
<td>30 Points</td>
</tr>
<tr>
<td>40 Points</td>
<td>40 Points</td>
<td>40 Points</td>
<td>40 Points</td>
<td>40 Points</td>
</tr>
<tr>
<td>50 Points</td>
<td>50 Points</td>
<td>50 Points</td>
<td>50 Points</td>
<td>50 Points</td>
</tr>
</tbody>
</table>
Write in Java:

let x: int ref ref = ref 1
int x = 1;
Write in Java:

let y (x: int): int =
    x * x + 1
int y(int x) {
    return x * x + 1;
}

back
Write in Java:

map

[3; 1; 2]

fun x -> x + 1
int[] a = [3, 1, 2];

for (int j = 0;
     j < a.length; j++) {
    a[j]++;
}

back
Write in Java:

```java
let new_counter ():
  (unit -> unit * unit -> int) =

let x: int ref = ref 0 in

(fun () -> x := !x + 1, fun () -> !x)
```
public class Counter {

    private int x;

    public Counter() { x = 0; }

    public void increment() { x++; }

    public int get() { return x; }

}
Write in Java:

type 'a tree =

| Empty

| Node of 'a tree * 'a * 'a tree
class Node<E> {

    // use null for empty tree

    public Node<E> left, right;

    public E element;
}

Give an example of data that can be modeled as ('a, 'b set) map
A classroom of students, each with a set of grades.
Give an example of data that can be modeled as `List<Map<E, F>>`
A bookshelf of phone books.
What is the main difference between OCaml and Java data structures?
Mutability.
What is one implementation strategy for the map ADT?
A set of key-value pairs, or a BST with no repeated elements and a key for each element.
What are two implementation strategies for the set ADT?
A list with no repeats, a BST with no repeats, or a map with dummy keys.
What are the static and dynamic types of x in the following?

```
Iterator<String> x = new Scanner();
```
Static type: Iterator<String>
Dynamic type: Scanner
Name two major differences between interfaces and abstract classes.
1. A class can extend one class but implement many interfaces.

2. An interface cannot contain implementation code; an abstract class can.
What is dynamic dispatch? How does it work in the abstract machine model?
Dynamic dispatch: what code (e.g. method body) is executed depends on the dynamic type of the object.

Type found by “pointer chasing” in the abstract model to determine the dynamic type.
List<C> l = new LinkedList<C>();
l.add(new A());
l.add(new B());
D d = l.get(0);

Draw an inheritance tree for classes A, B, C, and D.
List<String> i = new ArrayList<String>();
ArrayList<String> j =
    new ArrayList<String>();
List<Object> k;

For each, state whether valid. Assume they run independently.

i = j;
j = i;
k = i;
k = j;
\(i = j; \quad //\quad \text{Okay}\)
\(j = i; \quad //\quad \text{Bad}\)
\(k = i; \quad //\quad \text{Bad}\)
\(k = j; \quad //\quad \text{Bad}\)
What is a parent of all classes? Name two of its methods.
Object

String toString();
boolean equals(Object o);

(among others)
What is the difference between overriding and overloading? Which uses dynamic type, and which uses static type?
Overriding: redefining a method in a child class (same signature). Dynamic type.

Overload: methods with same name, but different arguments. Static type.
What does protected keyword mean?
Denotes a field that only the class and its subclasses can access.
Describe two classes in the I/O library, explain what operations you could perform with it.

Describe two different exceptions that may be thrown.
File – can open, create, delete files. Can throw FileNotFoundException.

Reader – (abstract class) allows you to read characters from a stream. Can throw IOExceptions (of which there are many flavors).
Name three different uses of the static keyword in Java; explain the effect on each kind of entity.
- **Classes**: static nested classes can be accessed without enclosing instance, e.g. `new Outer.Inner();` instead of `(new Outer()).Inner();`

- **Methods**: static methods belong to the class, e.g. `Math.cos(80);`

- **Variables**: static methods belong to the class, and are shared by all instances.
Complete the method:

// Returns twice the number if odd,  
// else the number if even.

int foo(int x);
int foo(int x) {
    if (x % 2 == 0) // even
        return x;
    else
        return x * 2;
}
Design classes/interfaces for the situation:

You are writing a Blackboard-alternative software for a class. The software has a dropbox for students to submit assignments (each assignment is either a text document, a code ZIP, or an image diagram). Assignments are identified by a numerical ID.

The professor wants to be able to:
- access all documents submitted for a given assignment, in the order they were submitted
- assign a grade to each assignment
interface Assignment {
    int getGrade();
    void assignGrade(int x);
    File getFile();
}

class Text extends Assignment ...
class Code extends Assignment ...
class Diagram extends Assignment ...

class DropBox {
    List<Assignment> getFilesForAssignment(int assignment_id);
}
Complete the method:

// Returns the number,
// or 0 if not a numerical string.
// You should use Integer.parseInt

int getIntFrom(String s);
int getIntFrom(String s) {
    int x = 0;
    try {
        x = Integer.parseInt(s);
    } catch (NumberFormatException e) {
        // Leave x = 0.
    }
    return 0;
}
Complete the method:

// Recursive method that returns true if string is palindrome.

boolean isPalindrome(String s);
boolean isPalindrome(String s) {
    if (s.length() <= 1)
        return true;
    if (s.charAt(0) == s.charAt(s.length()-1)) {
        String tmp = s.substring(1, s.length()-1);
        return isPalindrome(tmp);
    } else {
        return false;
    }
}
Complete the method:

// Reverse array in place.
void reverseArray(Object[] o);
void reverseArray(Object[] o) {
    for (int i = 0;
         i < o.length/2;
         i++) {
        Object tmp = o[i];
        o[i] = o[o.length-i-1];
        o[o.length-i-1] = tmp;
    }
}
Final Jeopardy

Make your wager
At the beginning of the year, we prescribed a design process to translate informal specifications into code. What are the four steps of the “design process”? 
1. Understand the problem.

2. Formalize the interface.

3. Write test cases.

4. Implement required behavior.