1. **Screen List**

In this section, the classes involved now include inheritance. If you look closely at the screens, some will have the same shape as others, but with an extra section added on to the bottom.

This represents the idea of inheritance, in which classes that extend other classes gain all the functionality of the parent class, along with adding new parts of their own.

1. **Inheritance Tree**

The inheritance tree button at the top will bring up a diagram showing the inheritance structure of the classes used in this question. If you are unsure what inherits from what, take a look at this diagram.

1. **Screen List, DropRegion**

Begin by dragging one of the variable types that inherit from another type into the central area.

(Transition on animation complete where gen = 2)

1. **Object List, DropRegion**

Now place an object whose type is the parent of the variable type you have placed.

(Transition on placement when object = parent of screen)

1. **ObjectList, Drop region**

As we can see, although the object fits into the screen for the variable type, it does not take up the entire space, and there are some slots in the screen that are empty. When the machine tries to get the information from these sections, there would be nothing there, causing it to error.

Similarly, in polymorphism, objects of a parent type cannot be used in place of objects of a child type, as they do not contain all the functionality of that child type.

(clear screen)

1. **ScreenList, Objectlist, DropRegion, ClearButton**

Now place a variable type of a class that is a parent of another class, along with an object that is of a class that inherits from the variable type.

(transition on placement when gen = 1 and other = child)

1. **DropRegion**

As we can see, the object now fully fits in the variable type, but has an extra section at the bottom. All required methods and fields of the parent class can be accessed, but those of the child will be ignored, as the machine follows the guide of the parent screen.

Similarly, in polymorphism, if a child object is passed to a variable type of a class that is the parent of that object, only the methods and fields defined in the parent can be accessed. Those in the child are ignored, as it treats it as an object of the parent.

Questions will now ask you about parent and child classes and the relationship between the two.