AP CS A – Karel J Robot Name: _____
Review Sheet Chapters 5-6 Date: _____ Per: _____

1. List the 8 primitive predicates that can be used in an `if` clause:

2. In order for a new class (subclass) that you write to be able to use the 8 primitives above, the class must have been derived from the _____ class at some point in the inheritance chain.

3. When is each of the following iteration structures used? (true/false – The choice of which loop construct to use (which tool) is an arbitrary decision left up to you and your preference.)
   a) while _____
   b) for _____

4. List the 4 steps for building a `while` loop. Also, in section 6.6, what are the 2 things we should ALWAYS check when writing any loop to help verify its correctness.
   _____
   _____
   _____
   _____
   _____
   _____

5. Write the correct form for the selection and iteration control structures. You should use generic statements such as `<test>` and `<instructions>`.
   a) for _____
   _____
   _____
   b) while _____
   _____
   _____
   c) if _____
   _____
   _____

6. simplify

```
if (frontIsClear())
{
        move();
}
else
{
        turnLeft();
        move();
}
```

7. simplify

```
if (nextToABeeper())
{
        move();
}
else
{
        move();
        turnRight();
}
```

8. simplify

```
if (!nextToARobot())
        return false;
else
        return true;
```

9. Complete the following class, adding any additional methods you might need to help you (note: you should at this point be able to recognize which methods need further modularization and which can be written only in terms of primitives - please modularize where appropriate).

```
public class JumpHarvestAndTest extends Robot {

    //precondition:  robot is facing east and standing next to the base of an arbitrarily high hurdle
    //postcondition: robot is facing east and is standing on the other side of the hurdle at the base
    public void jump()
    {









    }

    //precondition:  there are an arbitrary number of piles of beepers in a line terminated with a wall
                     segment – each pile has an arbitrary number of beepers x, such that 0≤x<infinity
    //postcondition: robot has picked up all beepers from the position it started at up to and including
                     the intersection next to the wall
    public void harvestAllToWall()
    {









    }

    //precondition:  robot has an arbitrary number of  beepers in bag
    //postcondition: returns true if rear is clear and the robot has exactly 2 beepers in its bag;
                     false otherwise
    //note:          you may NOT use &&  nor  || for this exercise
    public boolean rearIsClearAndRobotHas2BeepersInBag()
    {









    }
```

Friendly reminder: In order to help ensure the reading of the chapter, we usually put a question or 2 from the reading (which we didn't cover in class) on the test. :-)