Karel the Robot Reference Card

CPSC 231 - FALL 2018

built-in commands

```
move()
turn_left()
put_beeper()
pick_beeper()
```

These are the four primitive commands that Karel knows how to follow when he comes out of the box. Remember that the parentheses and the underscore character are part of the instruction.

program structure

```
"""
This is a comment for humans to read.
"""
from karel import *

⟨definitions of new instructions⟩

begin_karel_program()
⟨statements for Karel to execute⟩
end_karel_program()
```

This is the structure that every Karel program should take. You would save this code in a text file named with a `.py` extension and run it in our Karel Python environment. Any text sandwiched by `"""` lines or following a `#` symbol are comments for human readers, and are ignored by Karel.

defining new instructions

```
def ⟨new instruction name⟩():
    ⟨statements constituting the new instruction⟩
```

New instruction names start with a letter, and must be a continuous string of letters, numbers, and underscores. Every statement that belongs to the instruction is indented by four spaces.

conditions Karel can check

```
front_is_clear()
left_is_clear()
right_is_clear()
beepers_present()
beepers_in_bag()
facing_north()
facing_east()
facing_south()
facing_west()
# Test the opposite condition by writing "not" in front, e.g.:
not facing_north()
```

Karel can observe the state of the world, or his own state, but checking any of these nine conditions. At any given moment, each condition is either true or false. You can test the opposite of any of these conditions by writing `not` in front of it. Note the space, rather than underscore, between the word 'not' and the condition name.

conditional statements

```
if ⟨condition⟩:
    ⟨statements executed only if condition is true⟩

if ⟨condition⟩:
    ⟨statements executed only if condition is true⟩
else:
    ⟨statements executed only if condition is false⟩
```

Use a conditional statement when you want Karel to carry out a set of instructions only if a certain condition is true. Be careful to ensure that every instruction belonging to your conditional set is indented by exactly four spaces.

iterative statements

```
for i in range(⟨count⟩):
    ⟨statements repeated count times⟩

while ⟨condition⟩:
    ⟨statements repeated while condition is true⟩
```

Use an iterative statement when you want Karel to repeat a sequence of instructions a certain number of times, or while a certain condition is true. The ⟨*count*⟩ in the `for` statement must be a positive integer. Again, ensure that the group of instructions to be repeated are all indented by four spaces.