CSc 110 Final Exam
Cheat Sheet

```
def name(parameters):
    statement(s)
    ...
    return expression
```

```
for name in range(start, stop + 1, step):
    statement
    statement
    ...
    Statement
```

```
for item in collection:
    statement(s)
```

where ***collection*** is of type string, list, tuple or dictionary

```
while(condition):
    statements(s)
```

```
variable = type(input(prompt))
```

```
if (test):
    statement(s)
elif (test):
    statement(s)
else:
    statement(s)
```

Math class

Function name	Description
`ceil(`*value*`)`	rounds up
`floor(`*value*`)`	rounds down
`log(`*value, base*`)`	logarithm
`sqrt(`*value*`)`	square root

`sin(`*value*`)`
`cos(`*value*`)`
`tan(`*value*`)`
	sine/cosine/tangent of an angle in radians

`degrees(`*value*`)` `radians(`*value*`)`	convert degrees to radians and back

Constant	Description
`e`	2.7182818...
`pi`	3.1415926...

Other math functions:

Function name	Description
`abs(`*value*`)`	absolute value
`min(`*value1, value2*`)`	smaller of two values
`max(`*value1, value2*`)`	larger of two values
`round(`*value, i*`)`	Rounds value to the nearest *i* th digit.

String manipulation

	Description
s`.find(substring)`	Returns the index where the start of the given `substring` appears in string ***s*** (Returns -1 if not found.)
s`[index1: index2]`	Returns the characters in string ***s*** from index1 (inclusive) to index2 (exclusive); if index2 is omitted, grabs till end of string
s`.lower()`, ***s***`.upper()`	Returns a new string with characters in ***s*** converted to lowercase or uppercase letters
s`.startswith(substring)`	Returns `True` if ***s*** starts with string `substring` and `False` otherwise

`len(`**value**`)` – returns the length of **value**, where **value** may be a string, list, set or any collection type.

Declaring and using lists

name = `[`**value**`]` * **length**
name`[`**index**`]` = **value**

name.`append(`**value) –** append **value** to the end of list **name**
name.`pop(`**i**`)` – pop the item located at index **i** off of list **name**

Random

`randint(`**min**, **max**`)`

Sets

name = `set()`
name.`add`**(value)**

Dictionaries

name = `{}`
name`[`**key**`]` = **value**
value = **name**`[`**key**`]`

Operator `in`

item `in` **name** - returns `True` if **item** is in **name**; `False` otherwise

Classes

attribute (data inside each object)
`self`**.name = value**

method (behavior of each object)
`def` **name(**`self`**, parameters):**
statement(s)

Constructor(code to initialize new objects)
`def __init__(self):`
statement(s)

__str__ method(called when a string is required, as in print)
`def __init__(self):`
code produces a string

Inheritance

`class` **name(superclass)**

Critter classes

```
class name(Critter):
    def __init__(self):
        attributes (or instance variables)
    def eat(self):
        statement(s) that return True (eat) or False (don't eat)
    def fight(self, opponent):
        returns either ATTACK_ROAR, ATTACK_POUNCE, or ATTACK_SCRATCH
    def get_color(self):
        statement(s) that return a color as a string
    def get_move(self):
        statement(s) that return either DIRECTION_NORTH, DIRECTION_SOUTH,
        DIRECTION_EAST, DIRECTION_WEST, or DIRECTION_CENTER
    def __str__(self):
        statement(s) that return a string
```