HEROES Academy

Advanced Strings

Indexing, Slicing, Formatting, Extra Functions

```
target = "robots"
```

Indexing

```
## the index positions for "robots":
## r   o   b   o   t   s
## 0   1   2   3   4   5
## -6  -5  -4  -3  -2  -1

print(target[0])           ### prints the letter "r"

print(target[-1])          ### prints the letter "s"

print(target[5])           ### prints the letter "s"

first_letter = target[0]   ### first_letter now equals "r"
```

Slicing

```
## slicing has two possible numbers:  variable[start:stop]
## it will always go up to but not include stop

print(target[0:3]          ### prints "rob"

### the 0 is always implied if it's not there
print(target[:3]           ### prints "rob"

print(target[3:6])         ### prints "ots"

### the last index is always implied!
print(target[3:])          ### prints "ots"

### -1 indicates the last letter
### so the slice goes up to but doesn't include it!
print(target[:-1])         ### prints "robot"

### saves all but the first letter in a new variable
all_but_first = target[1:]
```

Concatenation

```
### the string “r”
first_letter = target[0]

### the string “obots”
all_but_first = target[1:]

### the string “obotsray”
piglatinified = all_but_first + first_letter + “ay”
```

Formatting

```
out_string = “The {} results were: \n\t{} correct \n\t{} wrong”

### we can prepare the string with the place holders (the {})
### notice the \n for a new line and \t for a tab!

print(out_string.format(“test”, 100, 0))
### prints
### The test results were:
###        100 correct
###        0 wrong

### When using this, don’t forget the following rules
### 1. You have to use the “.format()” on the strong
###    aka, “example string”.format()
### 2. the number of {} must match the number of values/variables
###    in format
### see the slides and website for more information!
```