Final exam points

Question type

- It is not multiple choice
- Read string transversal. Read how to index characters using square brackets []. Read string transversal using for and while loop

Print function

- `print(object(s), end=end)`

end: is an optional argument that specifies what to print at the end. The default is newline ‘\n’.

Example:

>>>print(‘hello ’, end = ‘world’)

>>>hello world

Making the third argument of the range function -1 allows backward iteration.

```
for i in range(5,0,-1):
    print(i)
```

Output:

5

4

3

2

1

String data-Type

- A string is a sequence of characters
- A string literal uses quotes 'Hello' or “Hello”
- For strings, + means “concatenate”
- When a string contains numbers, it is still a string

Traversing

Computations on string start at the beginning, select each character in turn, do something to it, and continue until the end. This pattern of processing is called a **traversal**.

Traversal: For loop

```
for letter in
'banana' :
    print(letter)
```

b	a	n	a	n	a

The iteration variable “iterates” though the string and the block (body) of code is executed once for each value in the sequence

Looking inside Strings

We can get at any single character in a string using an index specified in square brackets

b	a	n	a	n	a
0	1	2	3	4	5

```
>>> fruit = 'banana‘
>>> print(fruit[1])
```

indexing operator

The expression in brackets is called an **index**

The index value must be an integer and starts at zero

```
>>> fruit = 'banana'
>>> letter = fruit[1]
>>> print(letter)
a
>>> n = 3
>>> w = fruit[n - 1]
```

indexing operator

b	a	n	a	n	a
0	1	2	3	4	5

```
>>> print(fruit[-1])
```

Length of a string

- The len function, when applied to a string, returns the number of characters in a string

```
>>> fruit = "banana"
>>> len(fruit)
6
```

Traversal: While loop

```
fruit = ‘banana’


ctr = 0
while (
    print(
    ctr+=1
```

١٠

ctr < len(fruit)

fruit[ctr]

Slicing Strings

- A *substring* of a string is obtained by taking a **slice**.
- Need to get continuous section of a string using a colon operator.

Slicing

M	o	n	t	y		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11

```
>>> s = 'Monty Python'
>>> print(s[0:2])
Mo
```

print(s[:2])

Slicing

M	o	n	t	y		P	y	t	h	o	n
0	1	2	3	4	5	6	7	8	9	10	11

```
>>> print s[8:]
Thon
>>> print s[:]
Monty Python
```

A Character Too Far

```
>>> zot = 'abc'
>>> print(zot[5])
```

```
Traceback (most recent call last):
File "<stdin>", line 1, in <module> IndexError:
string index out of range
```

A Character Too Far

```
>>> zot = 'abc'
>>> print(zot[len(zot)])
```

```
Traceback (most recent call last):
File "<stdin>", line 1, in <module> IndexError:
string index out of range
```