Slicing

Extension to indexing

indexing is used for fetching a single character.
But slicing can do more.

```
name = "Varun"
print(name[:])
```

outputs to `Varun`

the `:` can be used for slicing
and can be achieved the same using `slice()`

- values are `[ {start} : {stop} : {step} ]`
  indexing works w.r.t `-1`.
- {start} is defaulted at `0` and is **inclusive**
- {stop} is needed to be specified till **one more** than required character and is **exclusive**
- {step} can be negative or positive, depending on the increment requirements.

defaultness of slicing

{start} is default at **starting point**
{stop} is **end of collection**
{step} is default at `+1`

Here's an example

```
name = "Varun"
print(name[0:2])
```

outputs to `Va`

Negative Slicing

```
name = "Varun"
print(name[-5:-1])
```

Forward printing

Using positive indexing

```
name = "Varun"
print(name[:])
```

`or`

```
name = "Varun"
print(name[0:5])
```

Both outputting to `Varun`

Using negative indexing

```
name = "Varun"
print(name[-5:])
```

outputting to `Varun`

Reverse printing

Using positive indexing

```
name = "Varun"
print(name[5::-1])
```

outputting to `nuraV`

Using negative indexing

```
name = "Varun"
print(name[-1::-1])
```

outputting to `nuraV`
