Print statements

In [30]:
```
print("My name is Om")
print("Hello World!")
```

```
My name is Om
Hello World!
```

Variable declaration

In [32]:
```
miles = 1000
name = "Om"
print(miles, "+", name)
while miles <= 1005 :
    print(miles)
    miles += 1

a, b, c = 15, 7.7, "Shree"
print(a, b, c)
```

```
1000 + Om
1000
1001
1002
1003
1004
1005
15 7.7 Shree
```

Typecasting

In [33]:
```
print(type(a))
print(type(b))
print(type(c))
print(56+100, 56*100, 56/100, 56-100)
fl = 100
print(float(fl))
print(str(fl))
print(type(str(fl)))
```

```
<class 'int'>
<class 'float'>
<class 'str'>
156 5600 0.56 -44
100.0
100
<class 'str'>
```

Lists

In [74]:
```
ls = ["om", 77, "shree", 56, 100.24, True]
print(ls)
print(ls[0], ls[2])
print(ls[1:3])
print(ls[1:])
print(ls * 2)
print(type(ls))
```

```
['om', 77, 'shree', 56, 100.24, True]
om shree
[77, 'shree']
[77, 'shree', 56, 100.24, True]
['om', 77, 'shree', 56, 100.24, True, 'om', 77, 'shree', 56, 100.24, True]
<class 'list'>
```

List operations

In [75]:
```
print(len(ls))
print("om" in ls, 23 in ls)
print(ls)
ls.insert(2, "Gautam")
print(len(ls))
print(ls)
```

```
6
True False
['om', 77, 'shree', 56, 100.24, True]
7
['om', 77, 'Gautam', 'shree', 56, 100.24, True]
```

Try for urself: max, min, compare, append, count, extend, remove, reverse, sort,

In [102]:
```
ls2 = [100, 200, 300, 77, 32, 420]
print(max(ls2))
print(min(ls2))
ls2.sort()
print(ls2)
print(ls == ls2)
ls2.reverse()
print(ls2)
print(ls2.count(77))
ls2.remove(32)
print(ls2)
ls2.append(ls)
print(ls2)
ls2.extend(ls)
print(ls2)
ls2.extend("Om")
print(ls2)
```

```
420
32
[32, 77, 100, 200, 300, 420]
False
[420, 300, 200, 100, 77, 32]
1
[420, 300, 200, 100, 77]
[420, 300, 200, 100, 77, ['om', 77, 'Gautam', 'shree', 56, 100.24, True]]
[420, 300, 200, 100, 77, ['om', 77, 'Gautam', 'shree', 56, 100.24, True], 'om', 77, 'Gautam', 'shree', 56, 100.2
4, True]
[420, 300, 200, 100, 77, ['om', 77, 'Gautam', 'shree', 56, 100.24, True], 'om', 77, 'Gautam', 'shree', 56, 100.2
4, True, 'O', 'm']
```

Dictionaries

In [113]:
```
student = {
    "name" : "Om",
    "roll" : 77,
    "cgpa" : 9.3
}
print(student)
print(student["roll"])
```

```
{'name': 'Om', 'roll': 77, 'cgpa': 9.3}
77
```

Nested dictionary

In [127]:
```
employ = {
    "Om" : {
        "name" : "Om",
        "roll" : 77,
        "department" : "management"
    },

    "Gautam" : {
        "name" : "Gautam",
        "roll" : 122,
        "department" : "development"
    },

    "random" : [123, True, False, "om", "Sakshi"]
}
print(employ)
print(employ["Om"])
print(employ["Om"]["roll"])
print(employ["random"])
print(employ["random"][4])
print(type(employ))
print(type(str(employ)))
```

```
{'Om': {'name': 'Om', 'roll': 77, 'department': 'management'}, 'Gautam': {'name': 'Gautam', 'roll': 122, 'depart
ment': 'development'}, 'random': [123, True, False, 'om', 'Sakshi']}
{'name': 'Om', 'roll': 77, 'department': 'management'}
77
[123, True, False, 'om', 'Sakshi']
Sakshi
<class 'dict'>
<class 'str'>
```

Tuples

In [142]:
```
tup1 = ("Om", 77, True, 100.56)
print(tup1)
print(tup1[2])
print(type(tup1))
```

```
('Om', 77, True, 100.56)
True
<class 'tuple'>
```

Mutable tuples

In [139]:
```
#Make a tuple mutable using typecasting to list
t1 = list(tup1)
t1.append("Hello")
tup1 = tuple(t1)
print(tup1)
```

```
('Om', 77, True, 100.56, 'Hello')
```

In [ ]: