The numeric data in Python is generally in three formats namely Integer, Float, and Complex.

We can create an integer variable using the two ways:

1. Directly assigning an integer value to a variable
2. Using a int() class.

In [ ]:

```
roll_no = 33
# display roll no
print("Roll number is:", roll_no)  # 33
print(type(roll_no))  # class 'int'

# create integer using int() class
id = int(25)
print(id)  # 25
print(type(id))
```

```
Roll number is: 33
<class 'int'>
25
<class 'int'>
```

In [ ]:

```
#data types
x=-9999
y=99.99
print(type(x))
print(type(y))
```

```
<class 'int'>
<class 'float'>
```

In [ ]:

```
#A Python complex number is one having a real and imaginary parts.
#Python provides a separate module called cmath with functions, especially for these comple

complex(1)
```

Out[4]:

```
(1+0j)
```

In [ ]:

```
complex(1,-2)
```

Out[5]:

```
(1-2j)
```

In [ ]:

```
x = 1    # int
y = 2.8  # float
z = 1j   # complex

#convert from int to float:
a = float(x)

#convert from float to int:
b = int(y)

#convert from int to complex:
c = complex(x)

print(a)
print(b)
print(c)

print(type(a))
print(type(b))
print(type(c))
```

```
1.0
2
(1+0j)
<class 'float'>
<class 'int'>
<class 'complex'>
```

Decimal uses ten digits, binary uses two digits, and hexadecimal uses sixteen digits. Since we only have the ten digits from the decimal system to use, we substitute letters for everything above the number nine. Therefore, the digits in hexadecimal are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F. They represent zero through nine, then A is worth ten, B is worth eleven, C is worth twelve, D is worth thirteen, E is fourteen, and F wraps up with a value of fifteen. So, there are sixteen digits in hexadecimal.

The eight digits of octal are 0, 1, 2, 3, 4, 5, 6, 7.

In [ ]:

```
i = 20
print("in binary form::", bin(i))
print("in octal form::", oct(i))
print("in hexa decimal form ::", hex(i))
```

```
in binary form:: 0b10100
in octal form:: 0o24
in hexa decimal form :: 0x14
```

###Type casting

Implicit Type conversion : Compatible data types are in general two numeric data types like int and float. It will convert the smaller data type to larger one to prevent any data loss. For eg.; int data will be converted to float.

In [ ]:

```
x = 99
y = 1.111

#data in int and float
print(type(x))
print(type(y))

#addition
z = x+y

#type after addition
print(type(z))
```

```
<class 'int'>
<class 'float'>
<class 'float'>
```

Explicit Type conversion : In Explicit Type Conversion, users convert the data type of an object to the required data type.

In [ ]:

```
int(123.654)
```

Out[13]:

```
123
```

In [ ]:

```
int("10")
```

Out[14]:

```
10
```

In [ ]:

```
int("10.5")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-16-54dd49a25c21> in <module>()
----> 1 int("10.5")

ValueError: invalid literal for int() with base 10: '10.5'
```

In [ ]:

```
int("ten")
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-15-846b3cf4a083> in <module>()
----> 1 int("ten")

ValueError: invalid literal for int() with base 10: 'ten'
```

In [1]:

```
#concatenating int and str types
i = 99
s = "apple"
print(s+str(i))
```

```
apple99
```

Python defines a set of functions that are used to generate or manipulate random numbers through the random module.

In [2]:

```
import random

# prints a random value from the list
list1 = [1, 2, 3, 4, 5, 6]
print(random.choice(list1))
```

```
1
```

In [3]:

```
import random

# using choice() to generate a random number from a
# given list of numbers.
print("A random number from list is : ", end="")
print(random.choice([1, 4, 8, 10, 3]))
```

```
A random number from list is : 8
```

In [4]:

```
print("A random number from range is : ", end="")
print(random.randrange(20, 50, 3))
```

```
A random number from range is : 38
```

In [ ]:

```

```