Second day python class

ECE

variables

- variable is to store the value
- nameing rules
    - should start with alphabets,underscroll(_), alphanumeric(should start with alphabets)
    - keywords and bullitents cannot be used as variable name.

code comments

- # single line
- '''text'''(or)"""text""" multiline

getting keyword list

In [1]:
```
import keyword
keyword.kwlist
```

Out[1]:
```
['False',
 'None',
 'True',
 'and',
 'as',
 'assert',
 'async',
 'await',
 'break',
 'class',
 'continue',
 'def',
 'del',
 'elif',
 'else',
 'except',
 'finally',
 'for',
 'from',
 'global',
 'if',
 'import',
 'in',
 'is',
 'lambda',
 'nonlocal',
 'not',
 'or',
 'pass',
 'raise',
 'return',
 'try',
 'while',
 'with',
 'yield']
```

operators

```
**-power
//-floor values(only int values)
```

membership operators(in,not in)

```
In [2]: a=[1,2,3,4,5,]
        if 5 in a:
            print(True)
```

```
True
```

identity operator(is,is not)

```
In [3]: a=10
        b=5
        if a is b:
            print(True)
        else:
            print(False)
```

```
False
```

Expressions

- operator precedency(PEMDAS rule)

```
In [4]: a,b,c,d=5,4,3,2
        print(a+b*c/d)
```

```
11.0
```

python literlas

- literals is a data which is given to the variables

types of literals

- string literals
  - single line ('',"")
  - multiline (""" """ or ''' ''')
- numeric literals
  - int, long, float, complex
- boolean and special literals
  - True, False, none
- literal collections
  - list, tuple, dictinory

```
In [5]: # single line
        a='ece'
        b="students"
        print (a,b)

        ece students
```

```
In [6]: # multiline literal
        a='''hai
        hello
        how r u bindu'''
```

```
In [7]: a

Out[7]: 'hai\nhello\nhow r u bindu'
```

```
In [8]: print (a
            )

        hai
        hello
        how r u bindu
```

reading user input

In [9]:
```
'''
a=34
print(a)
print(type(a))

a='hima'
print(a)
print(type(a))
'''
n= input('enter a value')
print(n)
print(type(n))
n= int(input('enter a value'))
print(n)
print(type(n))
```

```
enter a value

<class 'str'>
enter a value\

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-9-3b18e5c5e4cb> in <module>
     11 print(n)
     12 print(type(n))
---> 13 n= int(input('enter a value'))
     14 print(n)
     15 print(type(n))

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

Conditional statements

- used for decision making
- if the condition satisfies it just returns boolean values
- Types
  - if
  - else
  - elif

if statement

```
if condition followed by :
    statements to execute
```

ifelse stmt

```
if (condition):
    stmts to execute
else:
    stmts to execute
```

In [19]:

```
# valid user details or not
uname=input('enter u name:')
pwd=input('enter password:')
if uname=='hima' and pwd=='123':
    print('valid user details')
else:
    print('invalid')
```

```
enter u name:hima
enter password:123
invalid
```

syntax for if,elseif,elif statement

```
if (condition):
    stmts to execute
elif condition:
    stmts to execute
else:
    stmts to execute
```

In [25]:

```
## even r odd
n=int(input('enter number'))
if n%2==0:
    print(n,'is even')
else:
    print(n,'is odd')
```

```
enter number4
4 is even
```

In [1]:
```
###### elif
a=int(input('enter a value'))
b=int(input('enter b value'))
c=int(input('enter c value'))
if a==b==c:
    print('all are equal')
if a>b and a>c:
    print(a,'is biggest')
elif b>c:
    print(b,'is biggest')
else:
    print(c,'is biggest')
```

```
enter a value3
enter b value3
enter c value3
all are equal
3 is biggest
```

In [ ]:
```
# nested if
# elif
a=int(input('enter a value'))
b=int(input('enter b value'))
c=int(input('enter c value'))
if a==b==c:
    print('all are equal')
if a>b and a>c:
    print(a,'is biggest')
if b>c:
    print(b,'is biggest')
else:
    print(c,'is biggest')
```

In [ ]:

In [ ]: