Basic recursion

Copyright: Jagadeesh Vasudevamurthy

filename:basicrecursion.ipynb

All import here

In [ ]:
```
import sys # For getting Python Version
```

Basic recursion class

You will write code only here

In [ ]:
```
############################################################
# Recursion.py
# Author: Jagadeesh Vasudevamurthy
# Copyright: Jagadeesh Vasudevamurthy 2021
############################################################

############################################################
#  All imports here
############################################################

############################################################
#  class  test factorial
############################################################
class Recursion():
    def __init__(self):
        #Nothing can be added here
        pass

    ############################################################
    #          WRITE ALL YOUR PUBLIC FUNCTION BELOW
    ############################################################

    ############################################################
    #          WRITE ALL YOUR PRIVATE FUNCTION BELOW
    ############################################################

```

Recursion test class

NOTHING CAN BE CHANGED BELOW

In [ ]:

```
################################################################
# RecursionTest.py
# Test Bench for Recursion
# Author: Jagadeesh Vasudevamurthy
# Copyright: Jagadeesh Vasudevamurthy 2021
################################################################

################################################################
#  NOTHING CAN BE CHANGED IN THIS FILE
################################################################

################################################################
#  All imports here
################################################################
import sys # For getting Python Version
##from Recursion import *
from time import process_time

################################################################
#  class  test factorial
################################################################
class Test_fact():
    def __init__(self):
        self._test()

    def _test1(self,n:'int')->'void':
        o = Recursion()
        ans1 = o.factI(n)
        ans2 = o.factR(n)
        print("Fact(",n,") Iterative = ",ans1,sep="") ;
        print("Fact(",n,") Recursive = ",ans2,sep="") ;
        assert(ans1 == ans2)

    def _test(self):
        a = [0,1,5,10,20]
        for e in a:
            self._test1(e)

################################################################
#  class  test print asis
################################################################
class Test_print_asis_reverse():
    def __init__(self):
        self._test()

    def _test1(self,n:'int',reverse:'bool')->'void':
        o = Recursion()
        if (reverse):
            print(n, " in reverse order is as follows")
        else:
            print(n, " in asis order is as follows")
        print("Iterative")
        o.printI(n,reverse)
        print("Recursive")
        o.printR(n,reverse)

```

```
57      def _test(self):
58          a = [0,1,9,10,1986,1000, 1111,5267896714578]
59          for e in a:
60              self._test1(e,True)
61          for e in a:
62              self._test1(e,False)
63  
64  ##############################################################
65  #  class  reverse int
66  ##############################################################
67  class Test_reverse():
68      def __init__(self):
69          self._test()
70  
71      def _test1(self,n:'int')->'void':
72          o = Recursion()
73          ans1 = o.reverseI(n)
74          ans2 = o.reverseR(n)
75          print("Reverse(",n,") Iterative = ",ans1,sep="") ;
76          print("Reverse(",n,") Recursive = ",ans2,sep="") ;
77          assert(ans1 == ans2)
78  
79      def _test(self):
80          a = [0,1,9,10,1986,1000, 1111,5267896714578]
81          for e in a:
82              self._test1(e)
83  
84  ##############################################################
85  #  class  test fib
86  ##############################################################
87  class Test_fib():
88      def __init__(self):
89          self._test()
90  
91      def _test1(self,n:'int')->'void':
92          o = Recursion()
93          t1_start = process_time()
94          ans1 = o.FibI(n)
95          t1_stop = process_time()
96          d1 = t1_stop - t1_start
97          t1_start = process_time()
98          print("Fib(",n,") Iterative = ",ans1,"CPU =", d1) ;
99          ans2 = o.FibR(n)
100         t1_stop = process_time()
101         d2 = t1_stop - t1_start
102         print("Fib(",n,") Recursive = ",ans2,"CPU =", d2) ;
103         assert(ans1 == ans2)
104 
105     def _test(self):
106         #a = [0,1,9,10,14,20,40,45]
107         a = [0,1,9,10,14,20]
108         for e in a:
109             self._test1(e)
110         print("What CPU time it can take for  FibR(50)?")
111 
112 ##############################################################
113 #  class  Test permutation
```

```
114 ##############################################################
115 class Test_permutation():
116     def __init__(self):
117         self._test()
118 
119     def _test(self):
120         N = 5
121         for n in range(N):
122             o = Recursion()
123             print("Permutation of ",n, "are as follows");
124             a = o.permR(n)
125             print("Num permutation = ",a) ;
126 
127 
128 ################################################
129 #  class  Test Tower of Hanoi
130 ##############################################################
131 class Test_th():
132     def __init__(self):
133         self._test()
134 
135     def _test(self):
136         N = 6
137         for n in range(N):
138            o = Recursion()
139            print("Tower of Haonoi of ",n, "are as follows");
140            a = o.thR(n)
141            e = (1 << n) - 1
142            if (a != e):
143                print("Expected move is", e, "But you took", a , "moves")
144                assert(False)
145            print("Num Moves = ",a) ;
146 
147 ##############################################################
148 # test1
149 ##############################################################
150 def test1():
151     print("Basic Recursion test starts")
152     print(sys.version)
153     print("----------------Testing factorial--------------------")
154     t = Test_fact()
155     print("----------------Testing print asis/reverse--------------------")
156     t = Test_print_asis_reverse()
157     print("----------------Testing reverse--------------------")
158     t = Test_reverse()
159 
160     print("Basic Recursion test Passed. If you don't see this line means, yo
161 
162 ##############################################################
163 # test2
164 ##############################################################
165 def test2():
166     print("test2 starts")
167     print("----------------Testing Fibonacci--------------------")
168     t = Test_fib()
169     print("----------------Testing permutation by recursion----------------
170     t = Test_permutation()
```

```
    print("----------------Testing Tower of Hanoi by recursion--------------
    t = Test_th()
    print("test2 Ends")

################################################################
# MAIN
################################################################
def main():
    test1()
    test2()

################################################################
# start up
################################################################
if (__name__  == '__main__'):
    main()


```

In [ ]:
```
main()
```