```
from __future__ import division
import sys
import random
import math
import numpy as np
from models import *
from searchers import *
from options import *
from utilities import *
sys.dont_write_bytecode = True
#Dr.M
rand=  random.random # generate nums 0..1
any=   random.choice # pull any from list
sqrt= math.sqrt  #square root function


def step2():
   random.seed(14)
   test = MaxWalkSat("ZDT1")
   solution,score = test.evaluate()
   print "Solution: ",
   print solution
   print "Score: ",
   print score

   print myoptions

def display(modelName,searcher,runTimes,scores):
  assert(len(runTimes) == len(scores)),'Ouch! it hurts'
  print "============================================================"
  print "Model Name: %s"%modelName
  print "Searcher Name: %s"%searcher.__name__,
  print "Options Used: ",
  print myoptions[searcher.__name__]
  import time
  print ("Data: %s"%time.strftime("%d/%m/%Y"))
  for i in range(0,len(runTimes)):
    print "RunNo: %s RunTime: %s Score: %s"%(i+1,runTimes[i],scores[i])
  print scores
  print xtile(scores,width=25,show=" %1.6f")
  print "============================================================"



def multipleRun():
 r = 20
 for klass in [Schaffer, Fonseca, Kursawe, ZDT1]:
   #print "Model Name: %s"%klass.__name__
   for searcher in [SA, MaxWalkSat]:
     n = 0.0
     listTimeTaken = []
     listScores = []
     random.seed(1)
     for _ in range(r):
       test = searcher(klass.__name__)
       import time
       t1 = time.time()
       solution,score = test.evaluate()
       timeTaken = (time.time() - t1) * 1000
       listTimeTaken.append(timeTaken)
       listScores.append(score)
     display(klass.__name__,searcher,listTimeTaken,listScores)



if __name__ == '__main__':
 # random.seed(1)
 # nums = [random.random()**2 for _ in range(100)]
 # print xtile(nums,lo=0,hi=1.0,width=25,show=" %3.2f")
 # model = ZDT1()
 # model.testgx()
 # for klass in [ZDT1]:
 #   print klass.__name__
```



```
multipleRun()

```