```
"""
math_trainer.py
Train your times tables.
Initial Features:
* Print out times table for a given number.
* Limit tables to a lower number (default is 1)
and an upper number (default is 12).
* Pose test questions to the user
* Check whether the user is right or wrong
* Track the user's score.
Brendan Scott
February 2015
"""

#### Imports Section
import random
import sys  #H
import time #I

#### Constants Section
TEST_QUESTION = (4, 6)
QUESTION_TEMPLATE = "Combien font %s x %s ? "
LOWER = 1
UPPER = 12
MAX_QUESTIONS = 3 # for testing, you can increase it later
TIMES_TABLE_ENTRY = "%s x %s = %s"    #E1
TIMES_TABLE_ENTRY = "%2i x %2i = %3i " #E2, E3
INSTRUCTIONS = """Welcome to Math Trainer
This application will train you on your times tables.
It can either print one or more of the tables for you
so that you can revise (training) or it can test
your times tables.
""" #G
CONFIRM_QUIT_MESSAGE = 'Vraiment quitter (O/n)? ' #H
SCORE_TEMPLATE = "Ton score est de %s (%i%%) en %.1f secondes\n"

#### Function Section
def make_question_list(lower=LOWER, upper=UPPER, random_order=True):
    """ prepare a list of questions in the form (x,y)
    where x and y are in the range from LOWER to UPPER inclusive
    If random_order is true, rearrange the questions in a random order
    """
    spam = [(x+1, y+1) for x in range(lower - 1,upper)
                       for y in range(lower - 1,upper)]
    if random_order:
        random.shuffle(spam)
    return spam

def do_testing():
    """ conduct a round of testing """
    question_list = make_question_list()
    score = 0
    start_time = time.time() #I
    for i, question in enumerate(question_list):
        if i >= MAX_QUESTIONS:
            break
        prompt = QUESTION_TEMPLATE%question
        correct_answer = question[0]*question[1]
        # indexes start from 0
        answer = raw_input(prompt)

        if int(answer) == correct_answer:
            print("Correct !")
            score = score+1
        else:
            print("Incorrect, il fallait dire %s."%(correct_answer))

```

```
67
68  #H    print("Nombre de calculs corrects = %s"%score)
69      end_time = time.time()
70      time_taken = end_time-start_time
71      percent_correct = int(score/float(MAX_QUESTIONS)*100)
72      print(SCORE_TEMPLATE%(score, percent_correct, time_taken))
73
74  def display_times_tables(upper=UPPER): #F
75      # Display the times tables up to UPPER
76      tables_per_line = 6
77      tables_to_print = range(1, upper+1)
78      # get a batch of 5 or 6 to print
79      batch = tables_to_print[:tables_per_line]
80      # remove them from the list
81      tables_to_print = tables_to_print[tables_per_line:]
82      while batch != []: # stop when there's no more to print
83          for x in range(1, upper+1):
84              # this goes from 1 to 12 and is the rows
85              accumulator = []
86              for y in batch:
87                  # this covers only the tables in the batch
88                  # it builds the columns
89                  accumulator.append(TIMES_TABLE_ENTRY%(y, x, x*y))
90              print("".join(accumulator)) # print one row
91          print("\n") # vertical separation between blocks of tables.
92          # now get another batch and repeat.
93          batch = tables_to_print[:tables_per_line]
94          tables_to_print = tables_to_print[tables_per_line:]
95
96  # def do_quit():  #G
97  #     print("In quit")
98
99  def do_quit():    #H
100     """ quit the application"""
101     if confirm_quit():
102         sys.exit()
103     print("In quit (not quitting, returning)")
104
105 def confirm_quit():
106     """Ask user to confirm that they want to quit
107     default to yes
108     Return True (yes, quit) or False (no, don't quit) """
109     spam = raw_input(CONFIRM_QUIT_MESSAGE)
110     if spam == 'n':
111         return False
112     else:
113         return True
114
115 #### Testing Section
116 # question = TEST_QUESTION
117 # prompt = QUESTION_TEMPLATE%question
118 # correct_answer = question[0]*question[1] # indexes start from 0
119 # answer = raw_input(prompt)
120 # if int(answer)== correct_answer:
121 #     print("Correct!")
122 # else:
123 #     print("Incorrect")
124 #B question_list = make_question_list()
125 #B print(question_list)
126
127 #C for lower,upper in [(2, 5), (4, 6), (7, 11)]:
128 #C     question_list = make_question_list(lower, upper)
129 #C     print(question_list)
130
131 #D do_testing()
132 #F display_times_tables()
133
134
```

```
134 #### Main Section
135 if __name__ == "__main__":
136     while True:
137         print(INSTRUCTIONS)
138         raw_input_prompt = "Press: 1 for training,"+\
139                            " 2 for testing, 3 to quit.\n"
140         selection = raw_input(raw_input_prompt)
141         selection = selection.strip()
142         while selection not in ["1", "2", "3"]:
143             selection = raw_input("Please type either 1, 2 or 3: ")
144             selection = selection.strip()
145         if selection == "1":
146             display_times_tables()
147         elif selection == "2":
148             do_testing()
149         else: # has to be 1, 2 or 3 so must be 3 (quit)
150             do_quit()
151
```