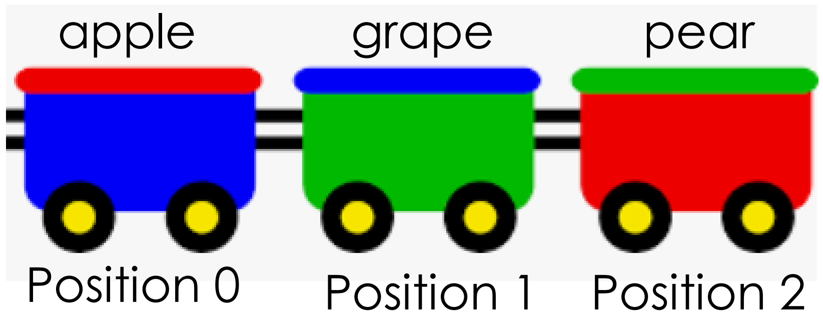
List data structure

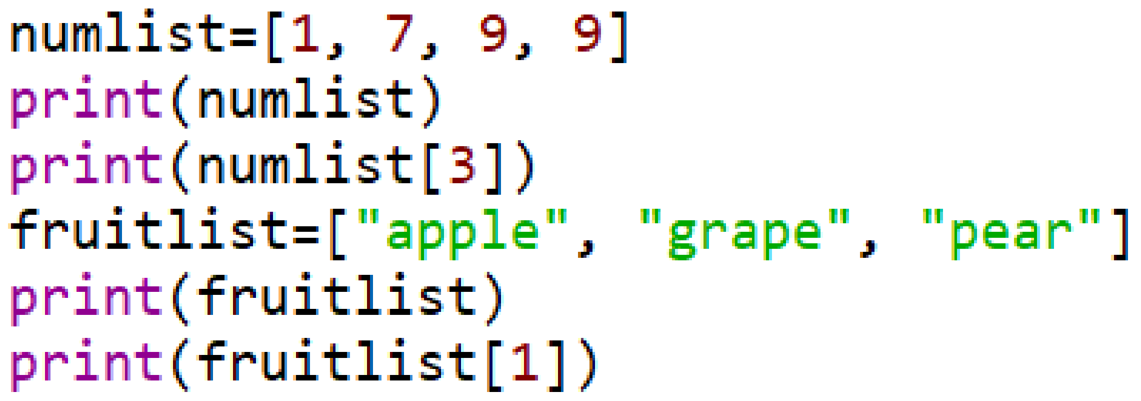
Python list

- List is a collection of data (numbers, string, etc.). Each data is associated with a position (indexed)
- Example:

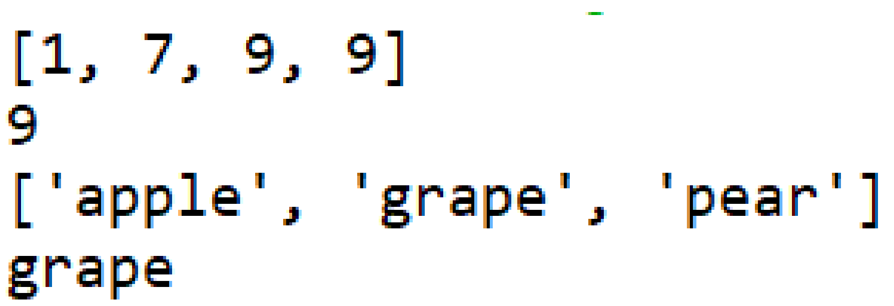
```
numlist = [1, 7, 9, 9]
fruitlist = [“apple”, “grape”, “pear”]
```



Example



```
numlist=[1, 7, 9, 9]
print(numlist)
print(numlist[3])
fruitlist=["apple", "grape", "pear"]
print(fruitlist)
print(fruitlist[1])
```



```
[1, 7, 9, 9]
9
['apple', 'grape', 'pear']
grape
```

List Method:	**Description:**
***list*.append(*x*)**	Adds item *x* to the end of the list
***list*.extend(*L*)**	Adds all items in list *L* to the end of the list
***list*.insert(*i*,*x*)**	Inserts item *x* at index position *i*
***list*.remove(*x*)**	Removes first item *x* from the list
***list*.pop(*i*)**	Removes item at index position *i* and returns it
***list*.index(*x*)**	Returns the index position in the list of first item *x*
***list*.count(*x*)**	Returns the number of times *x* appears in the list
***list*.sort()**	Sort all list items, in place
***list*.reverse()**	Reverse all list items, in place

Example

```
numlist=[1,12, 9, 9, 6, 9]
print("Number of 9 in the list is:\t", numlist.count(9))
print("The first position of 9 is:\t", numlist.index(9))
numlist.sort()
print("Sorted list is:\t\t\t", numlist)
numlist.reverse()
print("Reverse sorted list is\t\t", numlist)
```

```
Number of 9 in the list is:      3
The first position of 9 is:      2
Sorted list is:                  [1, 6, 9, 9, 9, 12]
Reverse sorted list is           [12, 9, 9, 9, 6, 1]
```

Example

```
fruitlist=["apple", "grape", "pear"]
vegalist=["lattuce", "carrot"]
fruitlist.append("peach")
print("The new list is:\t", fruitlist)
fruitlist.remove("grape")
print("After remove grage:\t", fruitlist)
tbe = fruitlist.pop(2)
print("Removed from the list:\t", tbe)
print("The new list is:\t", fruitlist)
fruitlist.extend(vegalist)
print("The extened list is:\t", fruitlist)
```

```
The new list is:        ['apple', 'grape', 'pear', 'peach']
After remove grage:     ['apple', 'pear', 'peach']
Removed from the list:  peach
The new list is:        ['apple', 'pear']
The extened list is:    ['apple', 'pear', 'lattuce', 'carrot']
```

Two methods to enumerate all elements in a list

```
fruitlist=["apple", "grape", "pear"]

for x in range(len(fruitlist)):
    print(fruitlist[x])

for x in fruitlist:
    print(x)
```

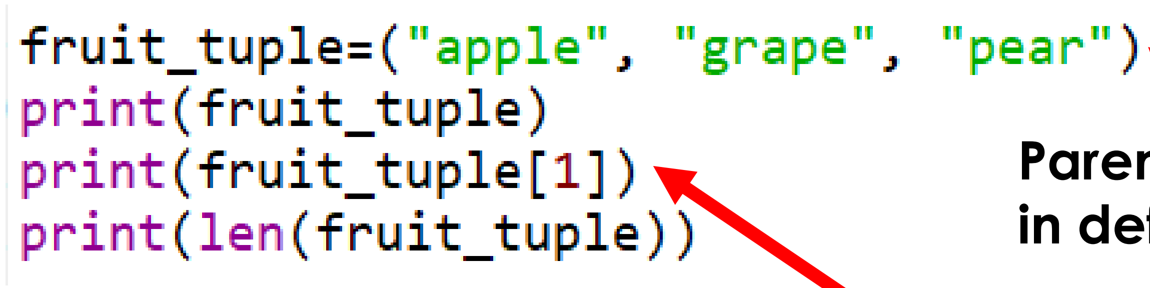
Method 1

Method 2

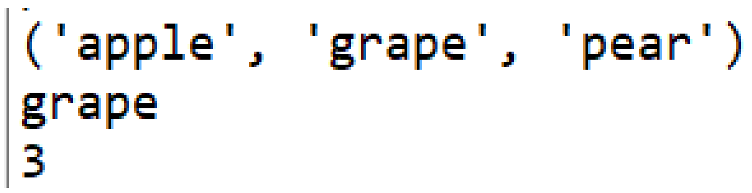
len(fruitlist) return the length of the list

Python tuple

- tuple is a collection of fixed data (their value cannot be changed
- Example:



```python
fruit_tuple=("apple", "grape", "pear")
print(fruit_tuple)
print(fruit_tuple[1])
print(len(fruit_tuple))
```



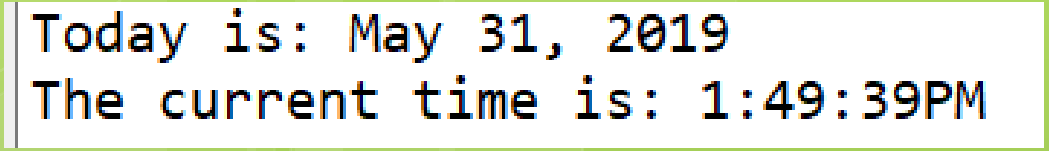
```
('apple', 'grape', 'pear')
grape
3
```

Parenthesis is used in defining a tuple

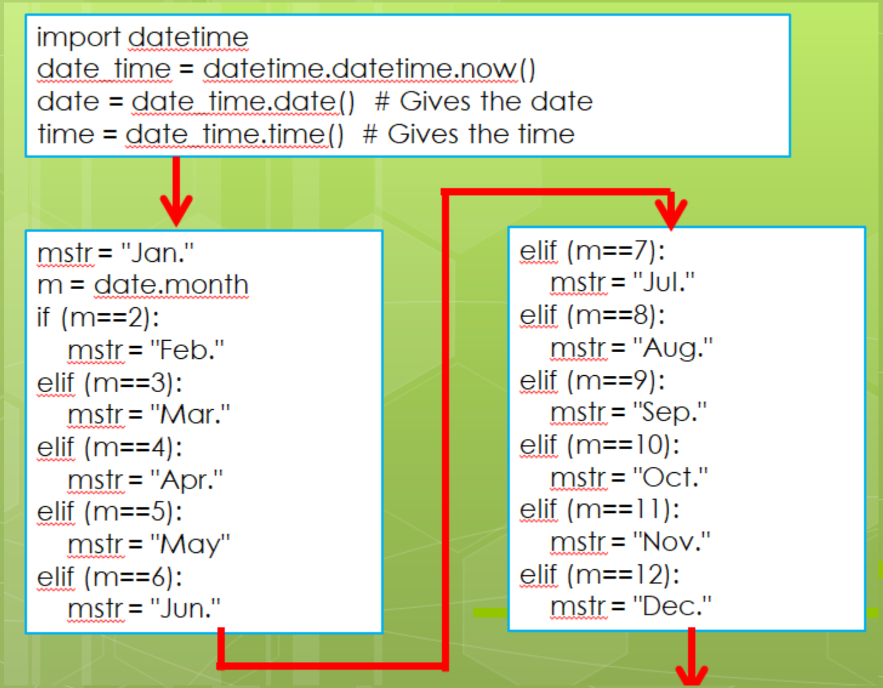
Square parenthesis is used in indexing

Today's Challenge 1

In previous program to display date, we used a long if-elif… statement to find the string for the current month. Can you use tuple data structure to simple the previous program?



```
Today is: May 31, 2019
The current time is: 1:49:39PM
```

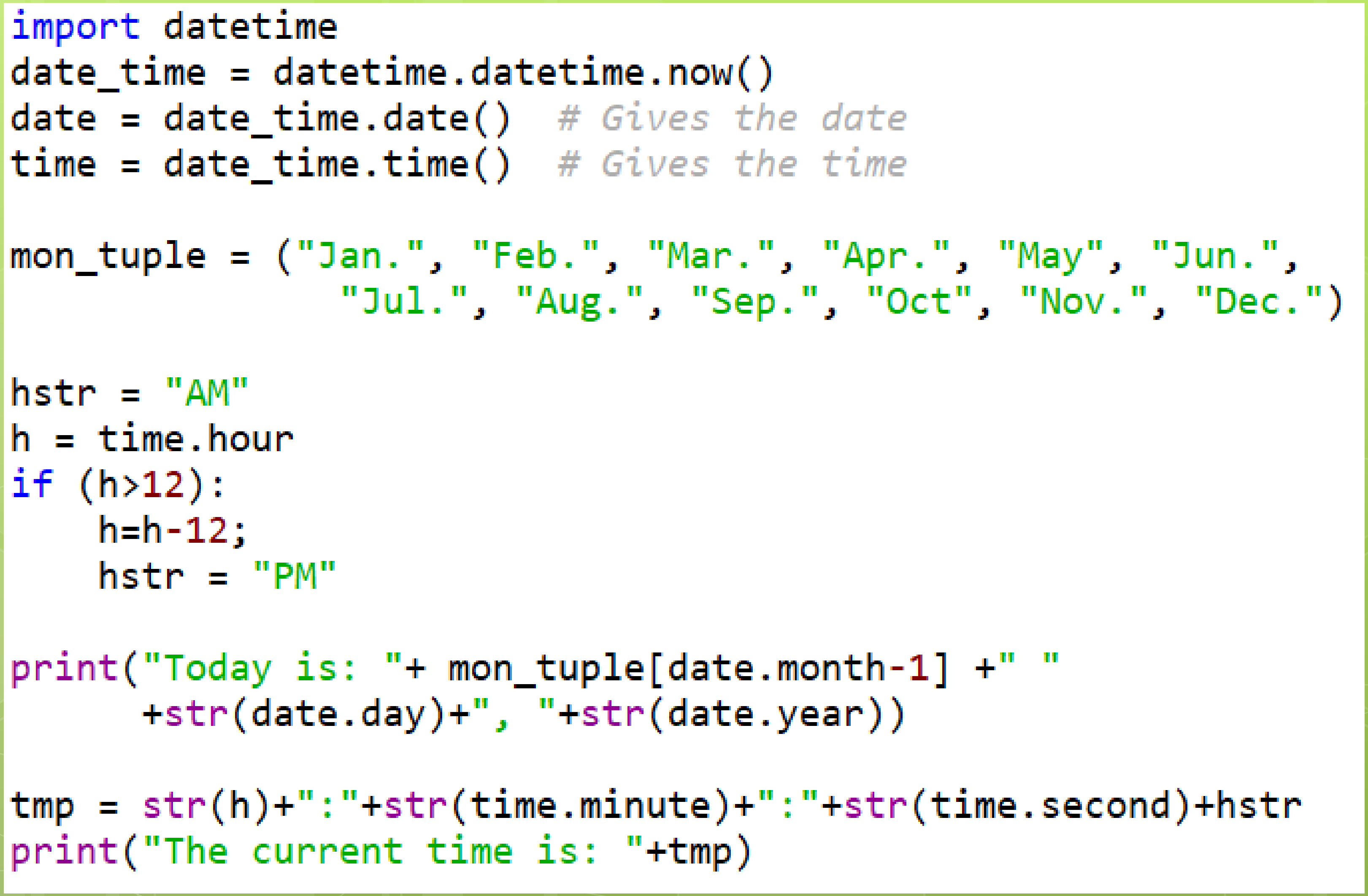


```
import datetime
date_time = datetime.datetime.now()
date = date_time.date()  # Gives the date
time = date_time.time()  # Gives the time
```

```
mstr = "Jan."
m = date.month
if (m==2):
    mstr = "Feb."
elif (m==3):
    mstr = "Mar."
elif (m==4):
    mstr = "Apr."
elif (m==5):
    mstr = "May"
elif (m==6):
    mstr = "Jun."
```

```
elif (m==7):
    mstr = "Jul."
elif (m==8):
    mstr = "Aug."
elif (m==9):
    mstr = "Sep."
elif (m==10):
    mstr = "Oct."
elif (m==11):
    mstr = "Nov."
elif (m==12):
    mstr = "Dec."
```

Solution for Challenge 1



```python
import datetime
date_time = datetime.datetime.now()
date = date_time.date()  # Gives the date
time = date_time.time()  # Gives the time

mon_tuple = ("Jan.", "Feb.", "Mar.", "Apr.", "May", "Jun.",
               "Jul.", "Aug.", "Sep.", "Oct", "Nov.", "Dec.")

hstr = "AM"
h = time.hour
if (h>12):
    h=h-12;
    hstr = "PM"

print("Today is: "+ mon_tuple[date.month-1] +" "
      +str(date.day)+", "+str(date.year))

tmp = str(h)+":"+str(time.minute)+":"+str(time.second)+hstr
print("The current time is: "+tmp)
```

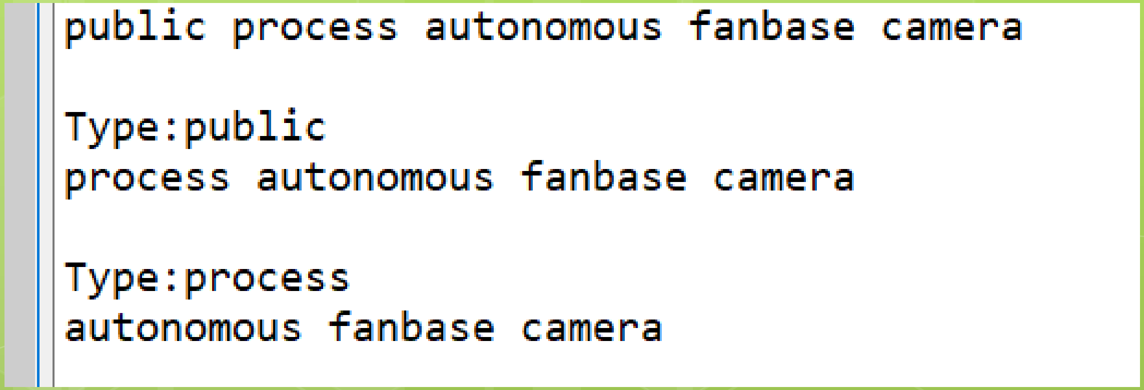
Today's Challenge 2

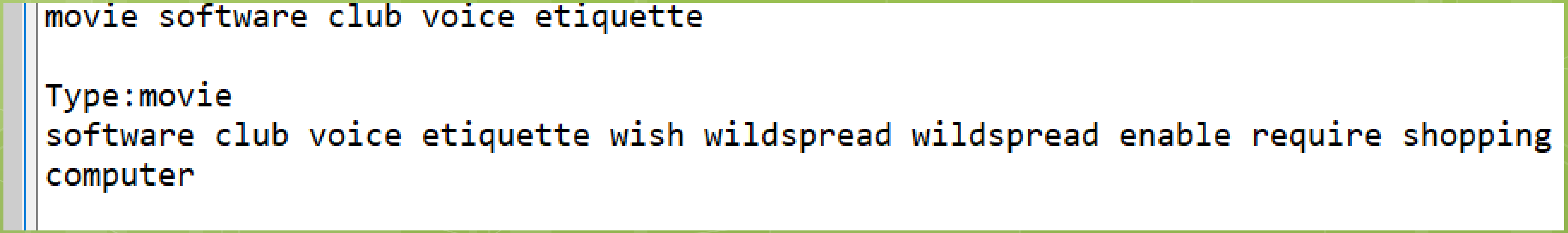
A typing game

- The program first display a list of 5 words
- If you type one of the 5 words correctly, the word will be removed from the list.
- The program also time how long you take to type, if you take longer than a specified time, the program will add new words to the list
- You win the game if there is no more words in the list
- You lose the game is there is more than 10 words in the list

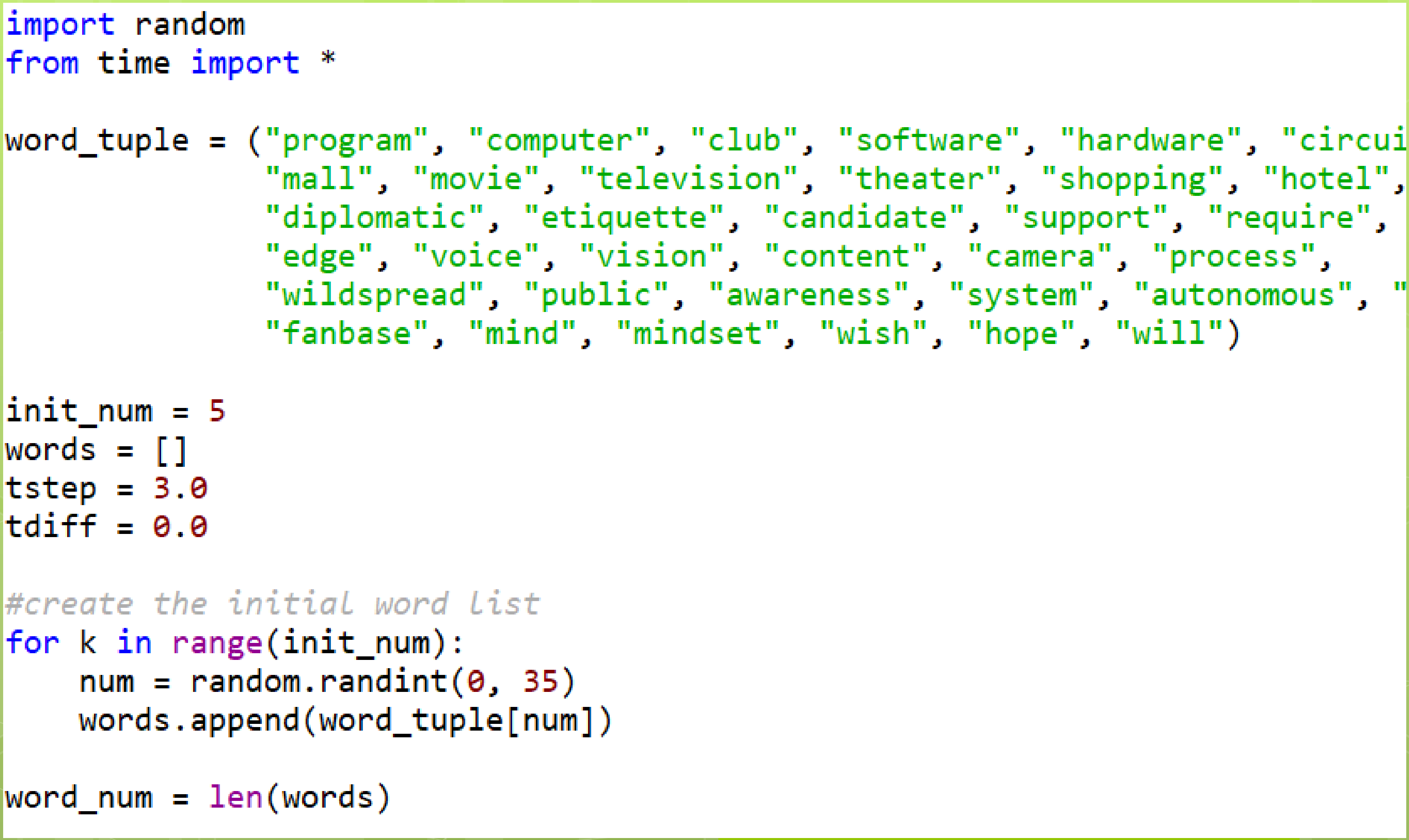
Today's Challenge 2

A typing game

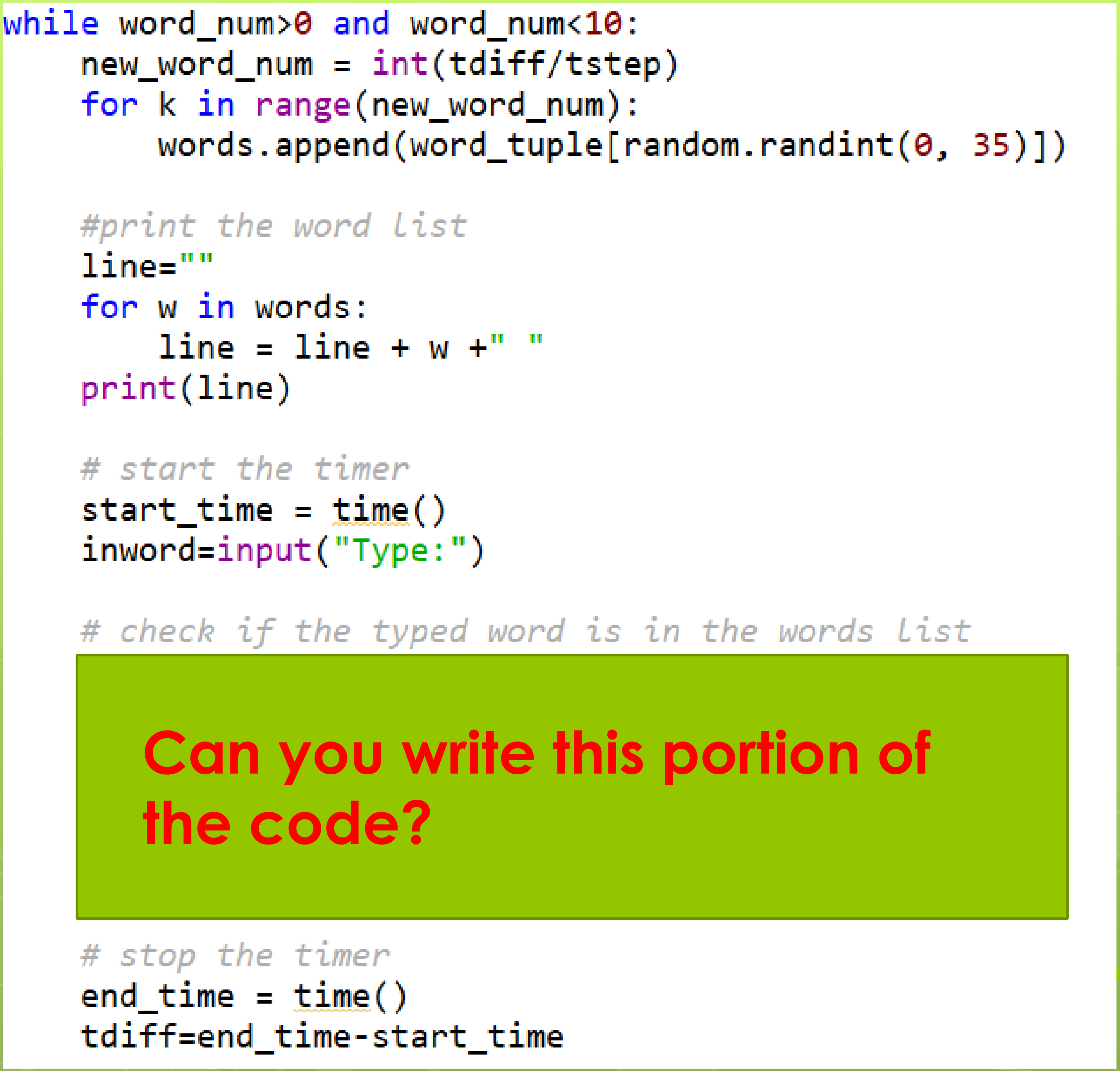

```
public process autonomous fanbase camera

Type:public
process autonomous fanbase camera

Type:process
autonomous fanbase camera
```


```
movie software club voice etiquette

Type:movie
software club voice etiquette wish wildspread wildspread enable require shopping
computer
```

A typing game



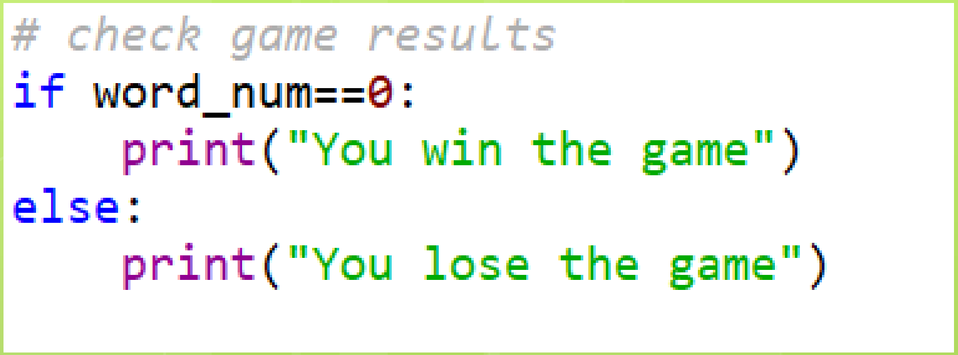
```
import random
from time import *

word_tuple = ("program", "computer", "club", "software", "hardware", "circui
              "mall", "movie", "television", "theater", "shopping", "hotel",
              "diplomatic", "etiquette", "candidate", "support", "require",
              "edge", "voice", "vision", "content", "camera", "process",
              "wildspread", "public", "awareness", "system", "autonomous", "
              "fanbase", "mind", "mindset", "wish", "hope", "will")

init_num = 5
words = []
tstep = 3.0
tdiff = 0.0

#create the initial word list
for k in range(init_num):
    num = random.randint(0, 35)
    words.append(word_tuple[num])

word_num = len(words)
```



```
while word_num>0 and word_num<10:
    new_word_num = int(tdiff/tstep)
    for k in range(new_word_num):
        words.append(word_tuple[random.randint(0, 35)])

    #print the word list
    line=""
    for w in words:
        line = line + w +" "
    print(line)

    # start the timer
    start_time = time()
    inword=input("Type:")

    # check if the typed word is in the words list
```

Can you write this portion of the code?

```
    # stop the timer
    end_time = time()
    tdiff=end_time-start_time
```



```
# check game results
if word_num==0:
    print("You win the game")
else:
    print("You lose the game")
```

- Can you find a bug of this program and take advantage of it for not losing the game (though you may not win)
- Can you remove the bug (debug)?