Bunny Races in Python3

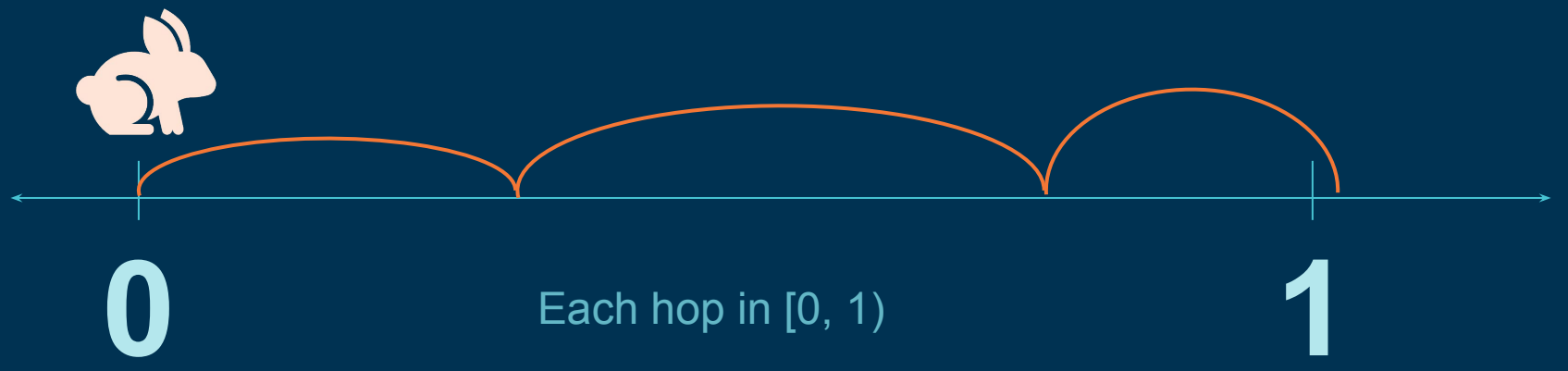
KarelCon 2021

Kent Pendleton

The Basics

Bunny Races in Python3

Welcome to the Bunny Races!



How many hops, on average, will it take the bunny to finish a race?

Expected Outcomes: Monitored and Summarized

```
===== Bunny Races! =====
How many bunnies to race? ('0' to quit): 5
Show each run? (y/n): y
1: 2
2: 4
3: 6
4: 2
5: 3
----- Summary Statistics -----
n = 5
Sum = 17
Average = 3.4
Minimum = 2
Maximum = 6
Frequencies:
[0, 2, 1, 1, 0, 1]
How many bunnies to race? ('0' to quit): 0

Thanks for using our app!
```

```
===== Bunny Races! =====
How many bunnies to race? ('0' to quit): 5000
Show each run? (y/n): n
----- Summary Statistics -----
n = 5000
Sum
```

Censored!

```
, 322, 167, 25, 6]
How many bunnies to race? ('0' to quit): 0

Thanks for using our app!
```

Python3 Coding Framework

Bunny Races in Python3

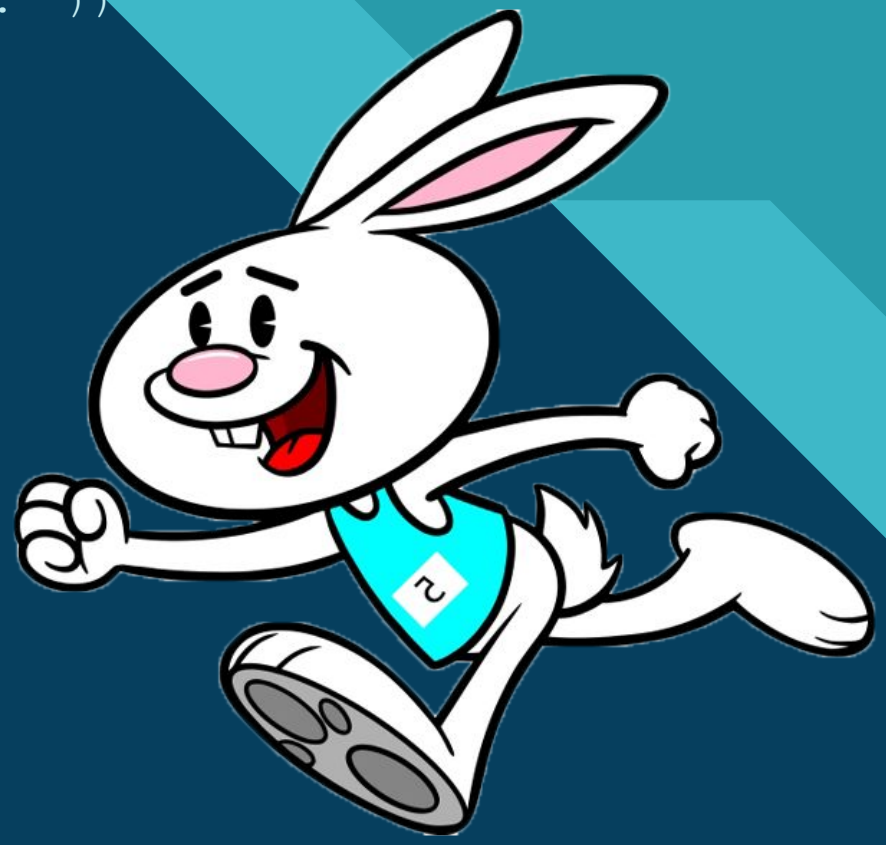
Python3 “Stub”

```
"""
Exercise:
Coder:
Date:
"""


#----- imports -----------------------------------------------------
#----- global variables --------------------------------------------
#----- my functions ------------------------------------------------
#----- main event --------------------------------------------------
#----- sample output -----------------------------------------------
```

Bunny Races: The 'main event'

```
print("===== Bunny Races! =====")
while True:
    n = int(input("How many bunnies to race? ('0' to quit): "))
    if n <= 0:
        break
    show_each_run = input("Show each run? (y/n): ")
    show_each_run = show_each_run.lower()[0]
    if show_each_run == "y":
        show_runs = True
    else:
        show_runs = False
    hops = []
    for b in range(1, n+1, 1):
        num_hops = race_a_bunny()
        if show_runs:
            print(str(b) + ": " + str(num_hops))
        hops.append(num_hops)
    #end for loop
    stats(hops)
#end while loop
print("\nThanks for using our app!")
```



Racing Secrets

Random Stuff

```
import random
a_hop = random.random()
#a float in [0, 1)
```

Function Stuff

```
def race_a_bunny():
    pass
#end function race_a_bunny

def stats(lst):
    pass
#end function stats
```

Statistical Stuff

```
Given: lst = [2, 3, 2, 7, 2, 9, 4, 3, 2, 3]
```

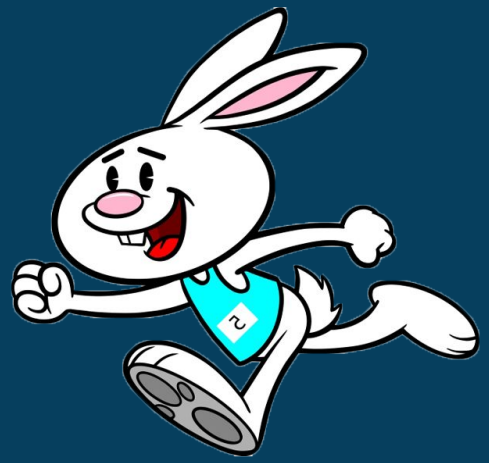
1. `n = len(lst) #len function: number of elements`
2. `a_sum = sum(list)`
3. `a_min = min(lst)`
4. `a_max = max(lst)`
5. `freq = [0] * a_max # create a list to house hops!`

Sometimes using built-in functions is easier than using ‘brute force’ but sometimes it’s also instructive to use ‘brute force’ to make sure you understand a process

Bunny Races: race_a_bunny function

```
def race_a_bunny():
    b = 0 #starting location for a bunny
    hops = 0 #accumulator
    while b < 1:
        hops += 1
        b += random.random() #gives numbers in [0, 1)
    #end loop

    return hops
#end function race_a_bunny
```



Bunny Races: stats function

```
def stats(lst):
    print("----- Summary Statistics -----")
    n = len(lst)
    print("n = " + str(n))
    a_sum = sum(lst)
    print("Sum = " + str(a_sum))
    an_avg = a_sum/n      #dwr! Data type?!
    print("Average = " + str(an_avg))
    a_min = min(lst) #built-in!
    print("Minimum = " + str(a_min))
    #For fun: find 'max' with brute force, not the built-in: a_max = max(lst)
    #assume the max is the first element in the list
    a_max = lst[0]
    for v in lst:
        if v > a_max:
        a_max = v
    #end for loop
    print("Maximum = " + str(a_max))
```

To be continued...

Bunny Races: stats function continued

```
def stats(lst):
    #frequency analysis
    print("Frequencies:")
    freq = [0]*a_max
    #iterate through lst, collecting frequencies
    for v in lst:
        freq[v-1] += 1
    #end for loop
    print(freq)
#end function stats
```

CodeHS!

https://codehs.com/sandbox/techtoolsguru/bunnyracespy3-start

- Avoid using variable names that are the same as function names
- Use meaningful, self-documenting variable names
- Never create a while loop that has no way of ending – provide a 'reachable' break
- Before doing something by 'brute force' ask: is there a built-in function that can do the job?
- It's 'good practice' to show that an app has ended (it's also very courteous!)
- If you ask for input, include a 'hint' as to what you want them to provide
- Assume the worst when users provide input and code accordingly
- When given a project, look to see if any sample output is provided
- Remember to str numeric output when necessary
- Watch out with *ANY* division use!
- Use meaningful comments
- Mark the end of major code blocks

Expected Outcomes: Monitored and Summarized

```
===== Bunny Races! =====
How many bunnies to race? ('0' to quit): 5
Show each run? (y/n): y
1: 2
2: 4
3: 6
4: 2
5: 3
----- Summary Statistics -----
n = 5
Sum = 17
Average = 3.4
Minimum = 2
Maximum = 6
Frequencies:
[0, 2, 1, 1, 0, 1]
How many bunnies to race? ('0' to quit): 0

Thanks for using our app!
```

```
===== Bunny Races! =====
How many bunnies to race? ('0' to quit): 5000
Show each run? (y/n): n
----- Summary Statistics -----
n = 5000
Sum = 13577
Average = 2.7154
Minimum = 2
Maximum = 7
Frequencies:
[0, 2478, 1702, 622, 167, 25, 6]
How many bunnies to race? ('0' to quit): 0

Thanks for using our app!
```

What we learn with pleasure, we never forget

- Alfred Mercier

https://codehs.com/sandbox/techtoolsguru/bunnyracespy3-final

Thank You!

Kent Pendleton
Email: kpendleton@trinitychristian.org

Visit
TNT:
technovicetools.com