Gauge Hartwell

18 January, 2021

CS427

Homework 4

1.

```python
import math
def main():
    # days = pow(2, 16)
    days = 356
    not_sharing = 1
    for i in range(1, days):
        print ("{0} - {1:.16f}".format(i+1, (1-(1-(i/days)))))
        if (i == 1000):
            break

if __name__== "__main__":
    main()
```

days = 356 for question A, days = pow(2, 16) for question B.

A) min(q | BirthdayProb(q, 356) ≥ 0.99) = 57
B) min(q | BirthdayProb(q, 2^{16}) ≥ 0.99) = 776

2.

```python
import random

def main():
    guesses = [] #stores the strings already generated
    iterations = 0 #number of guesses so far
    num = 0 #initialize byte value
    while (True):
        num = random.getrandbits(16) #random 16 bits
        if (num not in guesses): #check if num has already been generated
            guesses.append(num) #if not, append to guesses
            iterations += 1 #increment number of guesses
        else:
            break #break when collision occurs
    print (iterations)
    return

if __name__ == "__main__":
    main()
```

A) The rule of thumb for the expected samples before seeing a collision is sqrt(N). This is because collisions tend to become much more likely when q (the number of guesses) gets close to sqrt(N). Since we are randomly choosing 16-bit values, a collision is likely after 2^8 = 256

guesses.

B) Average of 20 runs: 274.65 fetches. Sqrt(65536) = 256 = 2^8. This is close to the average that I got.

3.

```python
import math
def main():
    cSet = 100 # change to match number of characters in charset
    numPass = pow(2, 128) # number of passwords needed
    i = 0 #represents the power that our charset is taken to in order to surpass numPass
    while (pow(cSet, i) <= numPass):
        i += 1
        print(i)
    return i

if __name__ == "__main__":
    main()
```

A) 28 characters w/ charset of 26 characters

B) 23 characters w/ charset of 52 characters

C) 22 characters w/ charset of 62 characters

D) 20 characters w/ charset of 100 characters