

# CS5200: Homework #1

Matthew Whitesides  
mbwxd4@mst.edu

September 1, 2019

1. Signed statement on last page.
2. Question 2.
  - (a) The age of the Earth is estimated at least 4.3 and commonly estimated at **4.54 billion years** old.
  - (b) The age of the Solar System is estimated between **4.53 and 4.58 billion years** old.
  - (c) The age of the Milky Way Galaxy is estimated at **11 to 13 billion years** old.
  - (d) The age of the universe is estimated at **10 to 15 billion years** old.
    - (Questions A - D): Dalrymple, G. Brent. *The Age of the Earth*. <https://pubs.usgs.gov/gip/geotime/age.html> Stanford University Press, 492p, 1991.
  - (e) The fate of the earth will probably be ended by our sun expanding into the earth in about 7 billion years putting the earth's lifespan at around **11 billion years** old.
    - Ward, Peter. *The life and death of planet Earth*. Holt, First edition, page 158, January 1, 2004.
  - (f) There are many ways earth could become uninhabitable by humans however ignoring man-made disasters, and unplanned events such as an asteroid, the earth will naturally go through temperature cycles that would destroy civilization. A man killing ice age is hard to predict but could begin anywhere from **one to tens of thousands of years** from now, or millions, the last one was only 11 thousand years ago. Recent human activities have increased the climate decay and instability, and to preserve the climate as long as possible we would need to reduce our carbon emissions, and possibly on the survival of the species perspective would eventually need to figure out ways to massively heat or cool the earth survive natural global temperature cycles.

- Revkin, Andrew C. *When Will the Next Ice Age Begin?*. <https://www.nytimes.com/2003/11/11/will-the-next-ice-age-begin.html> New York Times, Nov. 10, 2003, Section F, Page 6.
  - Reville, William. *How long will the human species survive on Earth?*. <https://www.irishtimes.com/news/science/how-long-will-the-human-species-survive-on-earth-1.2885564> Thu, Dec 8, 2016.
- (g) The estimated lifespan of the solar system will be about **11 billion years**. In around five billion years from now, the sun will enter a red giant phase and slowly expand outward for another billion years before exhausting all its energy.
- Redd, Nola Taylor. *Red Giant Stars: Facts, Definition and the Future of the Sun*. Schrder, Cannon Smith, R. (2008).
- (h) There are many theories about how the universe will end but a commonly used one is related to heat death. In anywhere from **1 to 100 trillion years**, there will not be enough matter/energy for stars to form and the universe will contain mainly black holes which themselves will disappear eventually.
- Wang, Yun. *RCurrent observational constraints on cosmic doomsday*. Journal of Cosmology and Astro-Particle Physics. 2004 (12)
- (i) If there are 31536000 seconds in a year then  $((2^{64} - 1) / 31536000)$  is equal to about **585 billion years** which if we go with the lower estimate of the universe lifespan is around **59%**.

### 3. Question 3.

- (a) Assuming the letters can be used more than once we'd have  $26^9$  combinations or **5.43 trillion**.
- (b) Using the same logic we'd have  **$K^9$  combinations**.
- (c) So if  $K^9 = 9000000000$  then  $K = \sqrt[9]{9000000000}$ , so  $K = 12.765$  or you'd need a **13 character** alphabet to work.
- (d)  **$K \times (K-1)^{n-1}$** . Pretty much it's the same except the first character is K possible letters and every other character is K - 1 possibilities.
- (e) Assuming we're going with small but not trying to min/max the use of paper too much, if we use 10px font, standard 8.5 x 11 letter size (not double sided), at 300px/inch printing, with a small margin of 2px between lines we'll get  $(11 \times 300)/12 = 275$  words per page. So **9 billion / 275 = 32.7 million pages**.
- (f) Looking at Amazon I see I can get Amazon Basics paper 5,000 sheets for \$44.99. So  $(32700000/5000) * 44.99 = 294234.6$  so it'd be a cool **\$294,234.60** assuming Amazon will sell us 6,540 boxes of paper. Now the best selling file cabinet I see on Amazon is a 3 drawer, 18 inch deep cabinet. Amazon lists the same paper in a 500 sheet ream as being 2 inches thick so we could get  $(250 \times 18) \times 3 = 13500$  per cabinet. So we'd need  $32700000/13500 = 2422.22$  or **2,423** cabinets.

- (g) They planned on about **15 thousand** years to finish the project. However they didn't say how many people would be working on it for that estimate so we can only assume it'd take one person that long. If they hope to finish the project in one persons lifespan given they start at 18 years old and retire at 60 years old (42 years per person) and assuming they accounted for the person working probably wouldn't work 24/7 in that original estimate, they'd need  $15000/42 = 357.14$  people (maybe a part time person for the 0.14).
- (h) The computer people estimated **one thousand** days (or 86400000 seconds) to finish, and saying there's 9 billion combinations (which isn't exact but they didn't give all details about their alphabet in the story) it'd have to do  $86400000/9000000000 = 0.0096$  **words/sec** which seems fast but maybe they print inline and on a large sheet with laser printers.
- (i) Good story, I like the Twilight Zone kinda ending. Although seems odd they'd be that afraid of Monks at the end. Does make you think about how little you use large numbers in everyday life and how futile it'd be to try to do an entire large task without breaking it down into small peices. Really anything over a 100 or so becomes nonsense in most peoples mind I feel.

4.

---

```
# Problem 1.
def altDif(n):
    if not n:
        return 0
    return n[0] - altDif(n[1:])

altDif([3, 5, 4])
# Out: 2
```

---

5.

---

```
# Problem 6
def f35qV2(n):
    if n < 8:
        return 'Error'
    elif 0 == n % 5:
        return (0, n / 5)
    elif 1 == n % 5:
        return (2, (n - 6) / 5)
    elif 2 == n % 5:
        return (4, (n - 12) / 5)
    elif 3 == n % 5:
        return (1, (n - 3) / 5)
    else:
        return (3, (n - 9) / 5)
```

---

```

# Output:
for i in range(8, 15):
    print(str(i) + ': ' + str(f35qV2(i)))

8: (1, 1.0)
9: (3, 0.0)
10: (0, 2.0)
11: (2, 1.0)
12: (4, 0.0)
13: (1, 2.0)
14: (3, 1.0)

```

---

6.

```

# Problem 7
def f(n):
    if n > 100:
        return n - 10
    else:
        return f(f(n + 11))

# Output
for i in [1, -6, 200, 27]:
    print(str(i) + ': ' + str(f(i)))

1: 91
-6: 91
200: 190
27: 91

```

---

So if you take away the second level of recursion and just try  $f(n + 11)$  it's easier to tell what's going on. The overall function won't return until we get a number higher than 100. So we'll keep calling  $(n + 11)$  until  $n$  is greater than 100. So for this simpler function  $n = 0$  will just keep adding 11 until we hit 110 after ten iterations (then the final answer will be  $110 - 10$  or 100) and then starting at 1 will also take 10 iterations to get to 111 and leave us with 101. However starting at 2 we'll get there in 9 steps giving us 91 and adding one until  $n = (11 + 2)$  or 13 where the process repeats until  $n$  is naturally greater than 100. Now when we add a second layer of recursion really we're just calling the deepest layer until it returns a number greater than 100 which happens first at  $(n = 2, 13, 24, 35, \text{etc})$  which is  $101 - 10$  or 91. So every  $n \geq 100$  is just 91.

```

def fV2(n):
    return 91 if n < 101 else n - 10

```

---

So given my explanation above how just  $f(n - 11)$  works  $f(f(n + 11))$  will always return 91 when the value is less than 101 and we know the rest

But really the simple rule is  $f(n)$  equals 91 when  $n < 101$  otherwise  $f$  equals  $(n - 10)$ .

[illegible]

8. \_\_\_\_\_

5

```

# Define GCD2
def GCD2(a, b, gcd = 0, s = 1, t = -1):
    if (a == b):
        return [a, 1, 0]

    if (gcd == 0):
        gcd = GCD(a, b)

    d = (a * s) + (b * t)
    if (d == gcd):
        #print("Returning:", gcd, s, t)
        return [gcd, s, t]

    if (d > 0):
        t -= 1
    elif (d < 0):
        s += 1

    #print("Returning GCD2(): ", a, b, gcd, s, t, "| d =", d)
    return GCD2(a, b, gcd, s, t)

# Results
print("Running GCD2 tests...\n")
rand_a_range = [random.randint(1,2500) for i in range(10)]
b = [random.randint(1,2500) for i in range(10)]
results = []

i = 0
for a in rand_a_range:
    x = GCD2(a, b[i]) # Call GCD2(a, b).
    y = (x[1] * a) + (x[2] * b[i]) # Validate the results.
    isValid = y == x[0]
    results.append([a, b[i], x, isValid, y])
    i += 1

print(tabulate(results, headers=['a', 'b', 'GCD2() = [g, s, t]', 'Validated', 'GCD'], tablefmt='orgtbl'))

```

Running GCD2 tests...

a	b	GCD2() = [g, s, t]	Validated	GCD
1619	216	[1, 107, -802]	True	1
360	240	[120, 1, -1]	True	120
2233	377	[29, 12, -71]	True	29
2315	16	[1, 3, -434]	True	1
2442	1746	[6, 143, -200]	True	6
788	1249	[1, 233, -147]	True	1
2143	580	[1, 367, -1356]	True	1
464	2280	[8, 172, -35]	True	8

1166   1418   [2, 377, -310]	True	2
286   1561   [1, 846, -155]	True	1

---

So essentially my thought process is, first handle the stopping conditions one when a and b are equal in which case you know either one can be 1 and the other 0. Then the real stop condition we get the GCD if we don't have it yet then stop when the gcd equals our calculation of  $(a * s) + (b * t)$ . Otherwise we check if we need to add to our negative counter if essentially if  $a - b$  is negative or we add to our positive counter. Then we just keep recursively calling till the calculation matches our gcd. Now it is pretty easy to high our recursion limit espically if the numbers have a small gcd and are large numbers. And to test I just generated some random numbers between 1 and 2500 for a and b and displayed the results.

9. 

---

```

# Problem 14
def SuperReverse(l):
    if len(l) == 0:
        return
    elif len(l) == 1:
        if isinstance(l[0], list):
            return [SuperReverse(l[0])]
        return l

    return SuperReverse(l[1:]) + SuperReverse(l[:1])

# Testing
l = ['1', '2', '3', '4', '5']
l2 = ['1', '2', ['3.1', '3.2', ['3.3.1', '3.3.2'], '3.4'], '4',
      '5']
print(SuperReverse(l))
print(SuperReverse(l2))

# Output
['5', '4', '3', '2', '1']
['5', '4', ['3.4', ['3.3.2', '3.3.1'], '3.2', '3.1'], '2', '1']

```

---

10. 

---

Anagram generator

---

11. Question 11

- (a) Australia.
- (b) Faculty meeting.
- (c) Philosophy department at the university of Walamaloo.
- (d) Five people, Michael and four Bruce's.
- (e) There is no rule #6.

---

*I Matthew Whitesides have read the syllabus carefully.*

---

A handwritten signature in black ink, appearing to be 'Matthew Whitesides', followed by a long horizontal line.

Thanks,  
Matt