Homework 0: Getting Started

Welcome to cs500! This is going to be a fun and exciting semester. Here are a couple of action items to complete to help you get started with the course. Feel free to reach out on EdStem if you have any questions!

1. The course website is found at `https://brown-cs500.github.io/spring-2026-website/`. This is where you will be able to find most of the information relating to the course, such as assignments and TA hours!

2. Check to see if you have been added to the course Gradescope where you will be submitting all of your (LaTeX-typeset!) assignments. If you have not been *automatically* added, send an email to the HTA and we will get you all sorted out. Please allow 48 hours from your enrollment in the course.

3. Join the course EdStem if you have not already! Be sure to check this frequently since this is where we'll be posting lecture materials, along with releasing FAQs/clarifications on any active homework assignment.

4. Finally, check out the course syllabus. To communicate, you have reviewed all course policies, and you agree to them.

If you have any feedback for the course, there is an *anonymous feedback form* located on the homepage of the course website. We would love to hear from you: both the good and (especially) the bad!

The rest of this homework includes some **practice problems**. These are only meant for you to check your preparation, **they are not meant to be turned in and shall not be graded**. Please pay particular attention to the last problem as it includes important information on the set up of the Python environment for the labs.

1 Syllabus Review

If you have not yet read the course syllabus, it can be found here. Please read through the syllabus, and proceed to answer the questions below.

Problem 1.

(a) What is the course late day policy?

(b) When is the latest you are allowed to get checked off for a lab?

(c) When are the midterm exam and the final exam?

(d) Consider the following situation: Alice and Bob meet up after class to talk about some of the concepts from the lecture that are on this week's homework assignment. They draw out some examples on a whiteboard and discuss some potential solutions. Bob points out that one of the examples they did was similar to one of the homework problems, and decided to do that problem together on the whiteboard and type up their solution directly from their shared notes before moving on. Do Alice's and Bob's actions violate the collaboration policy? Why or why not?

(e) Is using GitHub Copilot during a lab a violation of the collaboration policy?

(f) Suppose that, after the semester ends, a student makes their solutions available on GitHub to impress potential employers. Have they violated the course policy?

2 Preliminary Review Questions

To help test your preparedness for the mathematical content in this course, we have compiled a list of review questions.

2.1 Arithmetic

Recall the following exponentiation and logarithm rules:

$$\log a^b = b \log a \tag{1}$$
$$(b^x)^y = b^{xy} \tag{2}$$
$$b^x b^y = b^{x+y} \tag{3}$$

2.2 Modular Arithmetic

Recall the following modular arithmetic rules:

$$(a \cdot b) \bmod m \equiv (a \bmod m) \cdot (b \bmod m) \pmod m \tag{4}$$
$$(a + b) \bmod m \equiv (a \bmod m) + (b \mod m) \pmod m \tag{5}$$

Quotient Remainder Theorem: For any integers a, b, there exists q, r such that $a = qb + r$ where $0 \leq r < b$.

Problem 2.

(a) If $4^5 2^{17} = 2^x$, what is x?

(b) If $\log_2 2^3 8^7 = x$, what is x?

(c) What is $(17 \cdot 31) \bmod 7$?

(d) What is $(63 + 97) \bmod 9$?

(e) What integers q, r satisfy the quotient remainder theorem with $a = 39$, $b = 23$?

(f) What integers q, r satisfy the quotient remainder theorem with $a = 101$, $b = 11$?

2.3 Series

Problem 3. Write a closed-form formula for each of the following series.

(a)
$$3 + 5 + 7 + \cdots + (1 + 2n) + (3 + 2n)$$

(b)
$$\sum_{i=0}^{n-1} 2(n - i)$$

(c)
$$\sum_{i=0}^{n-1} (r^i), \text{ for } r \neq 1$$

(d)
$$\sum_{i=0}^{\log_2(n)-1} \frac{1}{4^i}$$

Solution. **Example:** Solving $1 + 2 + 3 + \cdots + n$

$$\begin{aligned} 1 + 2 + 3 + \cdots + n &= \sum_{i=1}^{n} i \\ &= \boxed{\frac{n(n+1)}{2}} \end{aligned}$$

□

2.4 Probability

Problem 4. For each of the following, show your work.

(a) Professor Rosencrantz flips a fair coin twice. Professor Guildenstern flips a fair coin once. What is the probability that Professor Rosencrantz obtains strictly more heads than Professor Guildenstern?

(b) You roll two ordinary, 6-sided dice.

(i) What is the expected value of the sum of the two values showing?

(ii) What is the expected value of the maximum of the two values showing?

(iii) How do these quantities change if instead of a 6-sided dice, we roll a 1000-sided dice?

(c) You flip a coin until you get your first tails. What is the the expected number of flips until your first tails?

(d) Consider two flips of a fair coin. Denote the outcomes by (X_1, X_2), where each X_i is Heads (H) or Tails (T). Define the following events:

$$A = \{\text{first flip is Heads}\}$$

$$B = \{\text{second flip is Heads}\}$$

$$C = \{\text{both flips match}\}$$

(i) Show whether A and B are independent or not.

(ii) Show whether A and C are independent or not.

(iii) Show whether B and C are independent or not.

(iv) Finally, check whether A, B, and C are **mutually** independent:

$$P(A \cap B \cap C) \stackrel{?}{=} P(A)\, P(B)\, P(C).$$

Explain your observations and reconcile any discrepancies you find.

(e) There are three pancakes in a stack. One is burnt on one side, one is burnt on two sides, and the other isn't burnt on either side. The pancakes are stacked on top of each other so that a burnt side is visible. What is the probability that the pancake that is burnt on both sides is on top?

3 Python Environment Set-up and Practice

Our labs will use Python, so you will need to set up some sort of Python environment to use for the lab assignments. We will follow the same structure for virtual environments as CS200:

1. Each lab will come with its own `requirements.txt` file that contains the requirements
2. You will be expected to create your virtual environment at the beginning of each lab. Instructions on how to do this can be found in the guide here.

Before your first lab, you should have the following installed:

(a) Python 3.10. You may use a more recent version of Python, but historically later versions have been buggy with the labs. If you don't yet have Python installed on your system, you can download it here.

(b) VS Code (which can be downloaded here). You may also use a different preferred IDE, but TAs will not be able help you regarding related issues (i.e., configuring pytest).

With these installed, and with the above virtual environment guide from CS200, you should be all set for your first lab!

3.1 Practice with Python

Python will be the language used for the labs. Here are some features that may be helpful. For slicing and list comprehension, format your code in LaTeX.

- **Exercise 1: Slicing**

  Python allows you to extract portions (subsequences) of lists, tuples, and strings using a concise *slicing* syntax.

  Listing 1: Basic Python Slicing

```
my_list = [10, 20, 30, 40, 50, 60, 70, 80]

# Slice from index 2 (inclusive) to index 5 (exclusive)
slice_1 = my_list[2:5]   # [30, 40, 50]

# Slice from the start until index 3 (exclusive)
slice_2 = my_list[:3]    # [10, 20, 30]

# Slice from index 4 (inclusive) to the end
slice_3 = my_list[4:]    # [50, 60, 70, 80]

# Slice with a step of 2
slice_4 = my_list[::2]   # [10, 30, 50, 70]

# Reverse a list using slicing
reversed_list = my_list[::-1]  # [80, 70, 60, 50, 40, 30, 20, 10]
```

  Using only a sequence of list slicing, from a list of the first 40 positive integers ($[1, 2, 3, \ldots, 40]$), select those which are of the form $5k+1$ where k is an integer, between 10 and 35 in descending order.

- **Exercise 2: List Comprehension**

  List comprehensions provide a concise way to construct lists in Python. They can often replace loops for building or transforming lists.

  Listing 2: Examples of List Comprehensions

```
# Generate squares of numbers from 0 to 9
squares = [x**2 for x in range(10)]  # [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

# Filter: keep only even numbers
evens = [x for x in range(10) if x % 2 == 0]  # [0, 2, 4, 6, 8]

# Convert a list of strings to uppercase
names = ["alice", "bob", "charlie"]
uppercase_names = [name.upper() for name in names]
# ['ALICE', 'BOB', 'CHARLIE']
```

  Using a list comprehension, create a list `cube_odds` that contains the cubes of all odd numbers between 1 and 20.

- **Exercise 3: Understanding your code**

  In Python, objects are assigned by object reference. For each of the following, what is the output of the program? (Try to think through these programs without running the code). These can be tricky, but will be helpful in helping you debug your labs.

  (a) **Direct Assignment to Same Object**

```
main_list = [0, 1, 2, 3, 4]
copy_list = main_list
copy_list[0] = 100
print(main_list)
print(copy_list)
```

  (b) **Slicing to Create a New List**

```
main_list = [0, 1, 2, 3, 4]
copy_list = main_list[:]
main_list[0] = 100
print(main_list)
print(copy_list)
```

  (c) **Reassigning a Slice Variable**

```
main_list = [0, 1, 2, 3, 4]
sub_list = main_list[1:3]  # [1, 2]
sub_list = [101, 102]
print(main_list)
print(sub_list)
```

  (d) **Referencing an Undefined Variable**

```
main_list = [0, 1, 2, 3, 4]
print(main_list)
print(sub_list)
```

(e) **Modifying a List of Lists**

```
main_list = [[0]] * 5
main_list[0][0] = 1
print(main_list)
```

(f) **Creating Distinct Nested Lists**

```
main_list = [[0] for _ in range(5)]
main_list[0][0] = 1
print(main_list)
```

(g) **String Copy and Concatenation**

```
list_str = "01234"
list_str_copy = list_str
list_str_copy = list_str_copy + "567"
print(list_str)
print(list_str_copy)
```

- **Other General Tips:**

1. Check out Python's Operator Precedence. Especially when dealing with bit operations, it is typically better to specify your order with parentheses to avoid unexpected bugs.