



March 29-30, 2025

UNC- Chapel Hill -Sitterson Hall

SOLHACKS

UNC Chapel Hill's First Hackathon for Latinos in Tech

Creating a welcoming and inclusive environment for Latinos in tech, fostering representation, building networks, and empowering innovation in the community



Students of all backgrounds



Beginner friendly



Sponsorship fair



Cool tech prizes

REGISTER NOW!



@solhacksunc



**Latinos
in Tech**

Hack110 Sign-Up Form!

When? Saturday, April 5th from 10 AM - 12 AM (Midnight)

Where? In Sitterson Lower Lobby

Who can join? Anyone in COMP 110! No prior experience required. Bring a partner or come as yourself (we'll have team-building activities if you want a partner)

Come for a fun day of coding, workshops and events (**food and CLE credit will be provided**):

- Choose between web development or game development track
- Go to various **workshops & events** such as: Navigating the CS Major, Resume workshop, ice cream station, and kahoot trivia and MORE!
- Link: Sign-Up Here! Or via the QR code 
- **Sign-Up form EXTENDED TO Monday, March 31st at 11:59 pm**
 - Spots are limited! So we'll prioritize interest!
 - If you have a partner, **ONLY ONE OF YOU** has to sign up - you will just enter your partner's info in the form.

Sign-Up Here!





CL23: Time Complexity (comparing lists and sets)

Announcements

- Quiz 02 grades will be released today – median ~85%!
- LS11 – Dictionaries due today
- EX03 released today, due *next Wednesday* (March 26)!
- Quiz 03 on Friday, March 28
 - Review Session on Wednesday (March 26) at 6:15pm in Fred Brooks (FB) 009

Warm-up diagram

```
1  def intersection(a: list[str], b: list[str]) -> list[str]:
2      result: list[str] = []
3
4      idx_a: int = 0
5      while idx_a < len(a):
6          idx_b: int = 0
7          found: bool = False
8          while not found and idx_b < len(b):
9              if a[idx_a] == b[idx_b]:
10                 found = True
11                 result.append(a[idx_a])
12                 idx_b += 1
13             idx_a += 1
14
15     return result
16
17
18 foo: list[str] = ["a", "b"]
19 bar: list[str] = ["c", "b"]
20 print(intersection(foo, bar))
```

After diagramming:

Assume our unit of "operation" is the number of times the block of lines #9-12 are evaluated.

Q1. Can different values of a and b lead to a difference in the number of operations required for the intersection function evaluation to complete?

Q2. If so, provide example item values for a and b which require the fewest operations to complete? Then try for the maximal operations to complete?

Q3. Assuming the item values of a and b are random and unpredictable, about how many operations does this function take to complete?

```
1 def intersection(a: list[str], b: list[str]) -> list[str]:
2     result: list[str] = []
3
4     idx_a: int = 0
5     while idx_a < len(a):
6         idx_b: int = 0
7         found: bool = False
8         while not found and idx_b < len(b):
9             if a[idx_a] == b[idx_b]:
10                 found = True
11                 result.append(a[idx_a])
12                 idx_b += 1
13             idx_a += 1
14
15     return result
16
17
18 foo: list[str] = ["a", "b"]
19 bar: list[str] = ["c", "b"]
20 print(intersection(foo, bar))
```

Comparing lists and sets

```
1 def intersection(a: list[str], b: list[str]) -> list[str]:
2     result: list[str] = []
3
4     idx_a: int = 0
5     while idx_a < len(a):
6         if a[idx_a] in b:
7             result.append(a[idx_a])
8             idx_a += 1
9
10    return result
```

```
1 def intersection(a: list[str], b: set[str]) -> set[str]:
2     result: set[str] = set()
3
4     idx_a: int = 0
5     while idx_a < len(a):
6         if a[idx_a] in b:
7             result.add(a[idx_a])
8             idx_a += 1
9
10    return result
```

Suppose a and b each had 1,000,000 elements, the worst case difference here is approximately 1,000,000 operations versus $1,000,000^2$ or 1,000,000,000,000 operations.

If your device can perform 100,000,000 operations per second, then...

A call to a will complete in 2.78 hours and b will complete in 1/100th of a second.