



CISSA PRESENTS

Foundations of Computing

Revision and Exam prep lecture

Today's Agenda



01

What to expect in the exam?

02

Tips and advice for preparation

03

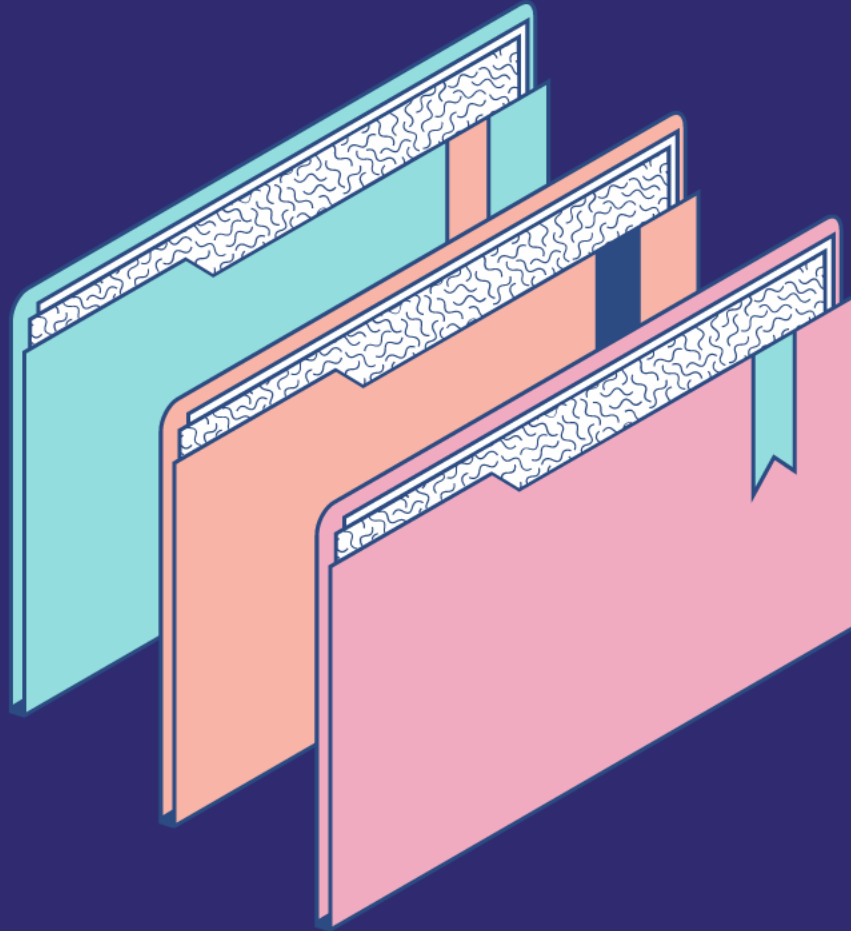
Solve practice questions in groups



Exam & Preparation Advice

with Cameron





Disclaimer

THIS IS NOT AN
EXHAUSTIVE GUIDE

- This lecture is meant to complement your prep
- We'll be looking at question types that have come up in the past
- You should prepare to see various other types or different iterations of these

Common type of questions

And how approach them.



Python one-liners

These test your ability to perform various small tasks.

Output value: 5

Required operations:

1. `range()`
2. `len()`
3. dictionary lookup

- Common ask you to do string, list or dictionary manipulation
- Tests your knowledge of common operations



Common Mistakes

- Not using the scratchpad

```
list('abcdefg').index('g')//2
```

- Using multiple lines

```
lst = list('abcdefg')  
lst.index('g')//2
```

- Printing results

```
print(list('abcdefg').index('g')//2)
```

- Using redundant operations

```
int(1+1), f"", bool(True), "word".lower()
```

- Not keeping it simple!

```
{'asdf': 'gfseafeasf', 'aaasdf': 'haef', 'aasdf': 'asfasdf'}['asdf'].index('g') + 6 // 2
```

Python one-liners

Some tips



Replacing loops

Given a block of code, you need to replace the for loop with while or vice versa

Rewrite the following function, replacing the **for** loop with a **while** loop:

```
def convert(s, n):  
    result = ''  
    for i in range(0, len(s), n):  
        v = ord(s[i]) - ord('a')  
        result += str(v)  
    return result
```


Ways to approach

- Be able to answer the following questions:
 1. What is the loop variable?
 2. Where does the loop variable start?
 3. How much does the loop variable increment by
 4. Where does the loop variable end?
- Careful with off-by-one errors!
`for i in range(0, len(s), n):` `while i <= len(s):`
- There's no excuse for not testing your code here
- Remember to practice reverse loops and nested loops

Replacing loops



Reorder jumbled code

Select and drag lines from here.

```
freqs[seq[i:i+k]] += 1
def common_ngrams(seq, k = 4):
    if seq[i:i+k] not in freqs:
        return common
    for i in range(0, len(seq)-k+1):
        common = []
        freqs = {}
        seq = seq.lower().strip()
        freqs[seq[i:i+k]] = 1
    else:
        for key, value in freqs.items():
            common.append(key)
            if value == _max:
                _max = max(freqs.values())
```

Construct your solution in this area.

- You'll be given a description and desired input/output
- Several lines of jumbled code which you need to put back in order
- Usually will contain functions, loops and conditional statements

Ways to approach

- Under how the input will flow through your code and reach the output
- Try and identify local structure

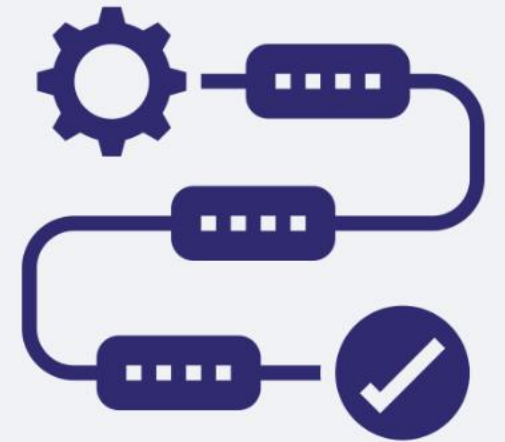
```
freqs[seq[i:i+k]] += 1
```

```
freqs = {}
```

```
def common_ngrams(seq, k = 4):
```

```
    for i in range(0, len(seq)-k+1):
```

Reorder jumbled code



Please please please indent!

```
def common_ngrams(seq, k = 4):
```

```
    freqs = {}
```

```
    seq = seq.lower().strip()
```

```
    for i in range(0, len(seq)-k+1):
```

```
        if seq[i:i+k] not in freqs:
```

```
            freqs[seq[i:i+k]] = 1
```

```
        else:
```

```
            freqs[seq[i:i+k]] += 1
```

```
    _max = max(freqs.values())
```

```
    common = []
```

```
    for key, value in freqs.items():
```

```
        if value == _max:
```

```
            common.append(key)
```

```
    return common
```

```
def common_ngrams(seq, k = 4):
```

```
    freqs = {}
```

```
    seq = seq.lower().strip()
```

```
    for i in range(0, len(seq)-k+1):
```

```
        if seq[i:i+k] not in freqs:
```

```
            freqs[seq[i:i+k]] = 1
```

```
        else:
```

```
            freqs[seq[i:i+k]] += 1
```

```
    _max = max(freqs.values())
```

```
    common = []
```

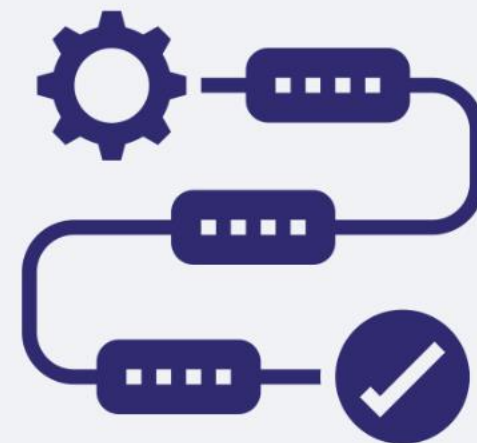
```
    for key, value in freqs.items():
```

```
        if value == _max:
```

```
            common.append(key)
```

```
    return common
```

Reorder
jumbled
code



Produce input / testcases

These require you to produce an input that will cause a specific output

```
(a) Valid row that is correctly classified
(b) Invalid row that is correctly classified
(c) Invalid row that is *in*correctly classified
(d) Valid row that is *in*correctly classified
```

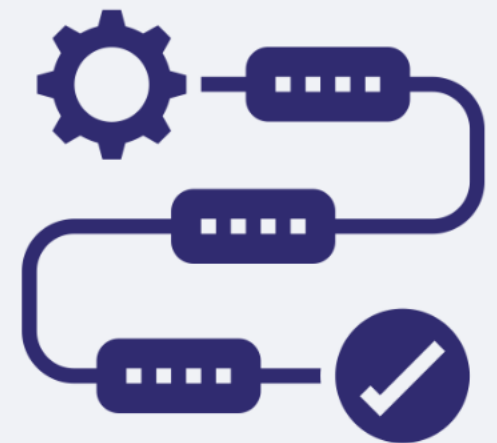
- You are typically given buggy code
- You will be given a description and details description of input
- You need to come up with inputs / testcases that produce certain results

Ways to approach






- Understand what's wrong with the code
- Try and think of edge case inputs
- If all else fails just try *something!*

| | Output True | Output False |
|----------------|-------------|--------------|
| Code Correct | 1 | 2 |
| Code Incorrect | 3 | 4 |

Produce specific input



Fill-in the blanks

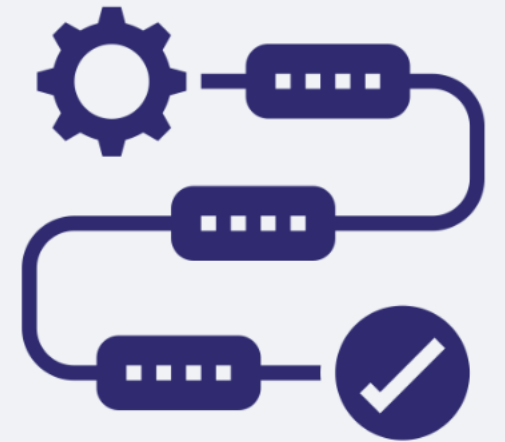
```
def steady(prices):  
      
    steadies = 0  
    changes = 0  
      
    if p == 'r':  
          
        changes += 1  
          
        overall -= 1  
        changes += 1  
    else:  
        steadies += 1  
    
```

- Given a description and inputs/ outputs, fill in parts of the code.
- Majority of the times involves a completing a function
- Usually some easier blanks to fill

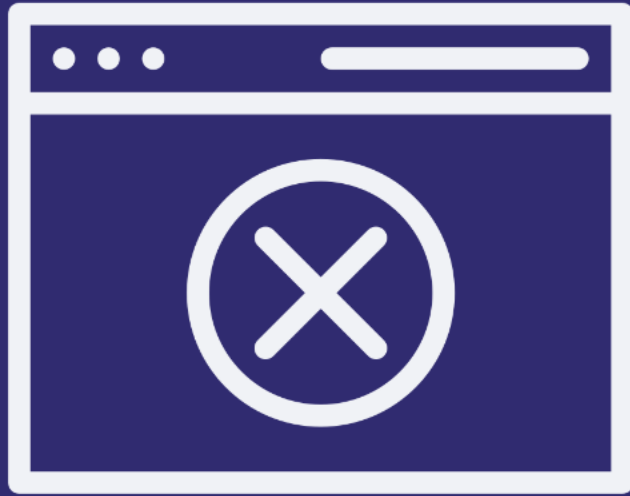
Ways to approach

- The code is usually broken into chunks. Understand what that specific block is responsible for.
- Look out for any specific clues/edge cases
- Pay attention to indentation
- If you have time it may be worth copying the question into the scratchpad

Fill in the blanks



Identify the Error



- Given buggy code:
 - Find 3 Errors
 - Identify Error
 - Give Fix
- Typically 4/5 errors

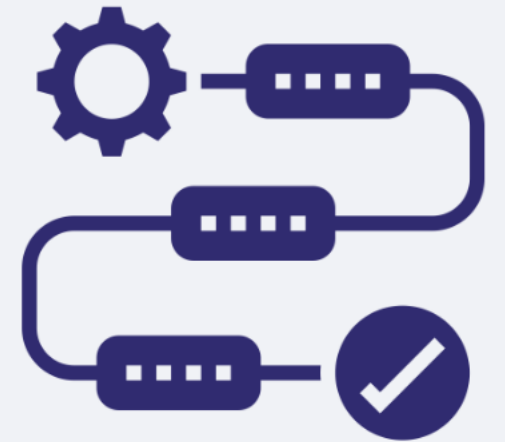
Ways to approach

- Look for Syntax errors first!
- Know the difference between Syntax and Runtime

```
num == 12  
zero = 0  
ans = num / zero  
print(ans
```

- If you find more than 3, submit the ones you're most confident in

Identify the Error



Writing a function

- Write a function to do XYZ
- Sample input out is given
- Usually is the last (or longest question)



Ways to approach

- Understand what exactly is being asked. If it's a wall of text, it usually contains hints.
- These type of problems can seem complex at first: Try to break down the problem into smaller steps.
- Try to solve one thing at a time. Incremental progress.
- Run your code! Don't lose marks to Syntax errors!

Writing a function



Any Questions?

FEEL FREE TO ASK



Preparing for the exams



How I prepared



Practice the Grok sheets

They're an easy way to brush up on fundamentals

Do past exam papers

https://library.unimelb.edu.au/examination_papers

Solve lots of problems

Google "Project Euler" and "Hackerrank"

Tips during the exam



...

During the exam



Avoid exploring the web too long

It's okay to look up syntax but you're unlikely to find a full solution.

Have notes and cheatsheets

It's much easier to Ctrl+F for things you know you have.

Play by your strengths

Any strategy is better than no strategy

Common Mistakes





Common Mistakes



Time management

It's easy to loose track of time.

Misunderstanding the question

Reading and understanding questions are key.

Underutilisation of tools

Various techniques to increase speed and accuracy.

Thank you for
listening ❤️

Please ask any
questions you have
:)

Good
Luck
Y'all!
