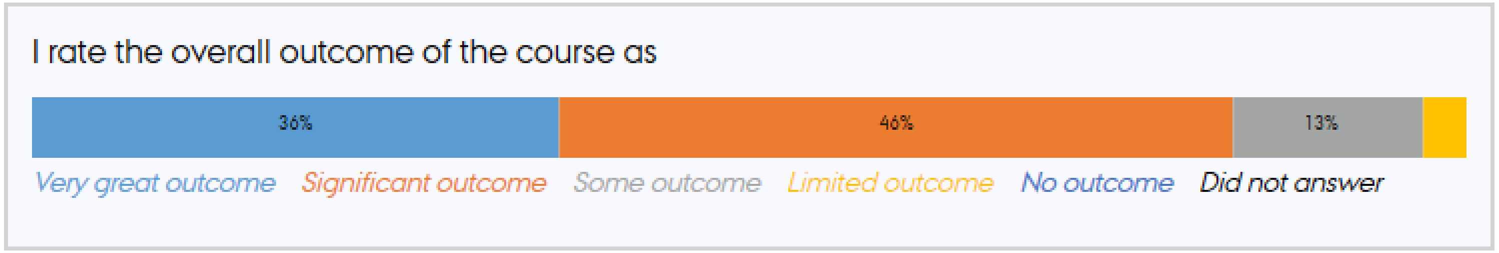
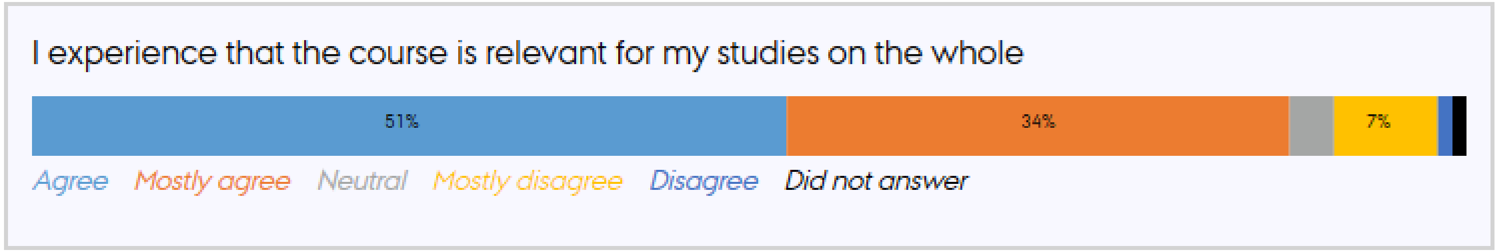
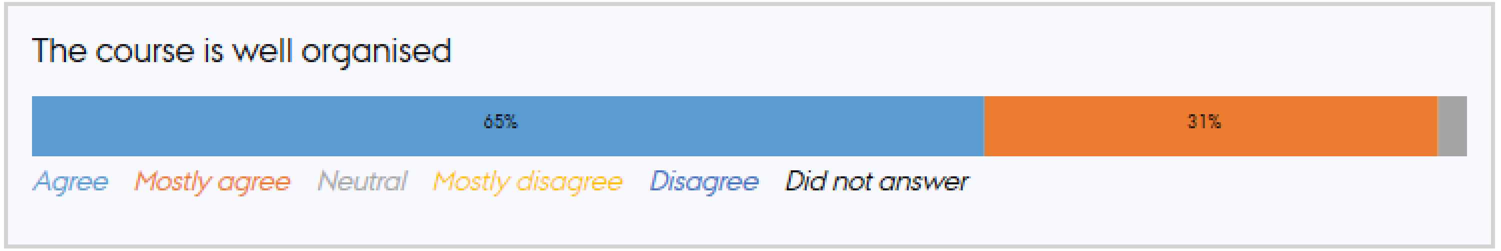
Introduction to Programming with Scientific Applications

- Missing handins – deadline: ASAP, but latest May 31st
- Final project – deadline: May 31st
  - June 1st the exam office is informed who passed the mandatory course assignments
- Course evaluation
- Exam
- AOB

Course evaluation



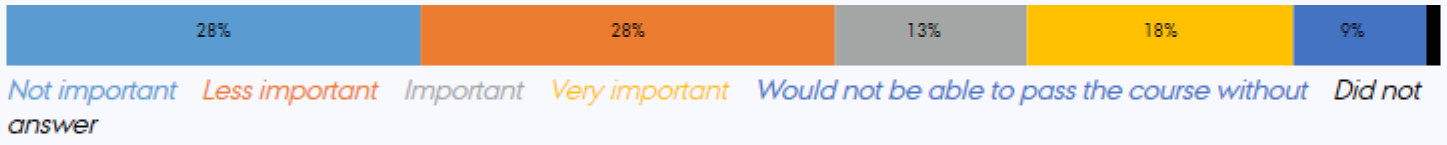




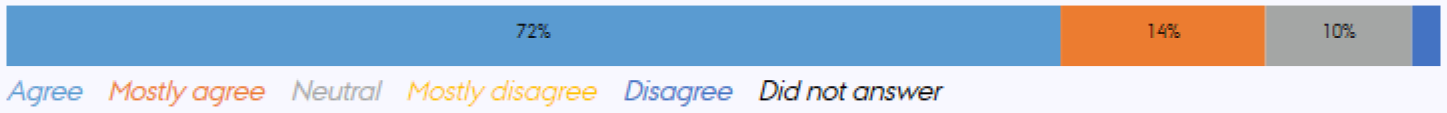
- ”Not that relevant for my math studies... but will be super relevant when entering the real world”
- Videos sometimes delayed

Your background

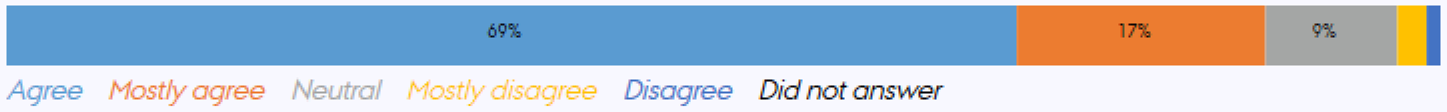
How important was it to you that the course was a study café ?



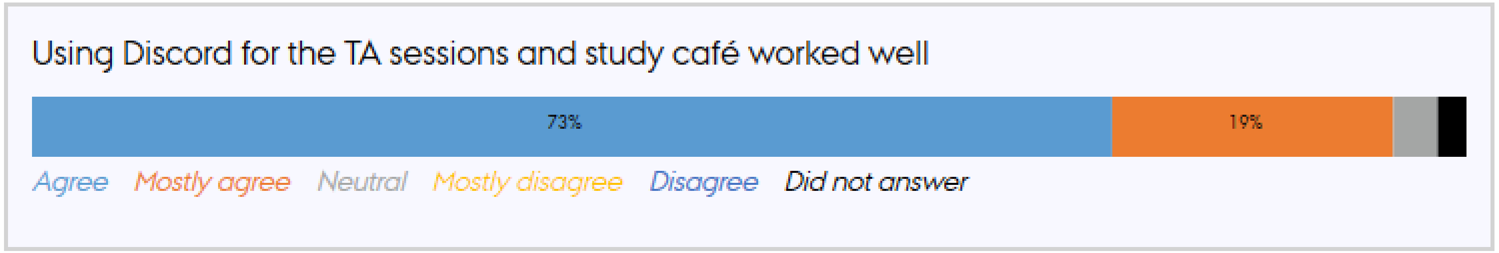
It was important that were several options for the final project

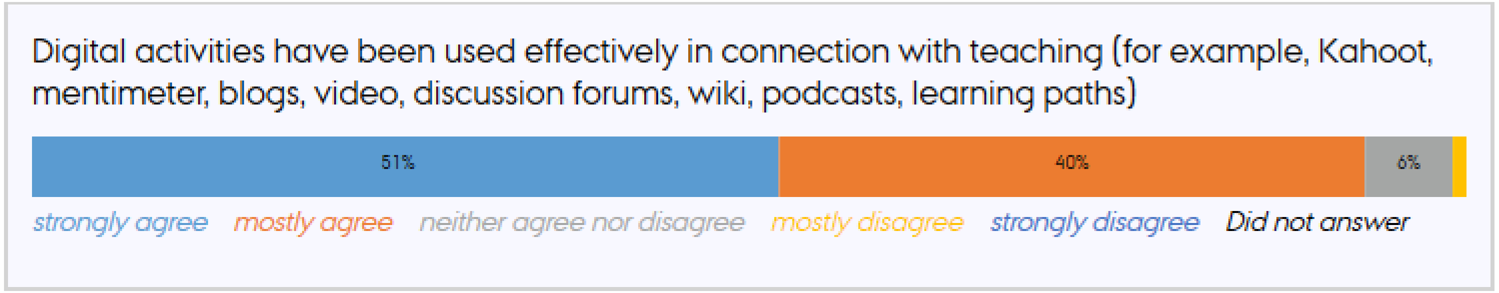


I had sufficient mathematical bagground to follow the course



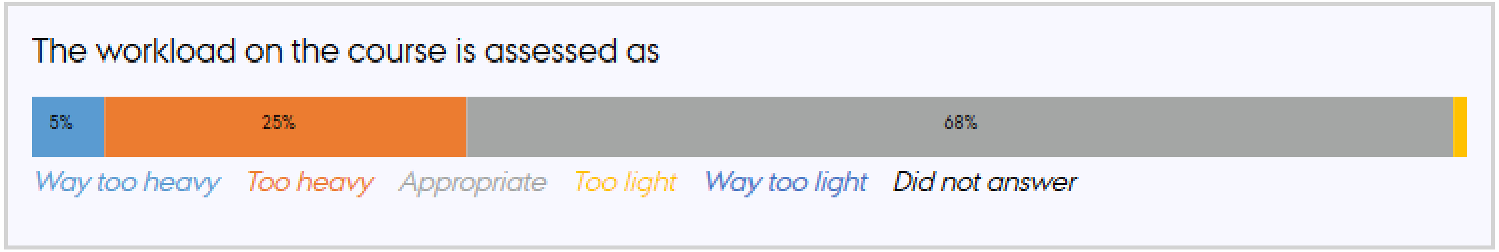
Digital activities

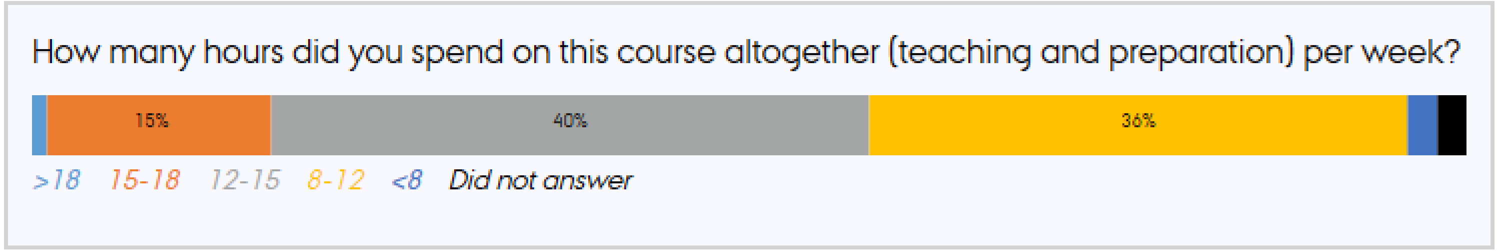


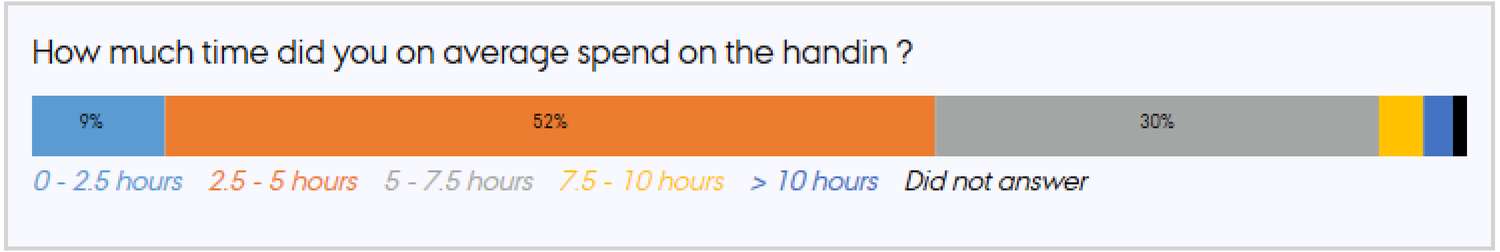


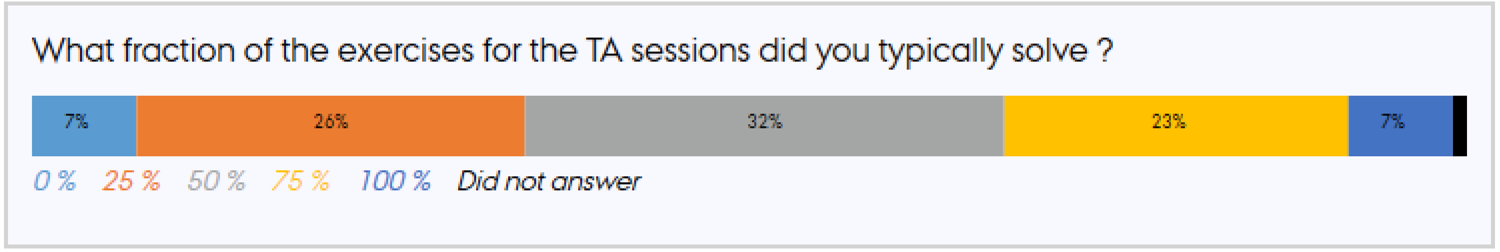
- Would have used the study café more, if it had been on discord from the beginning

Workload



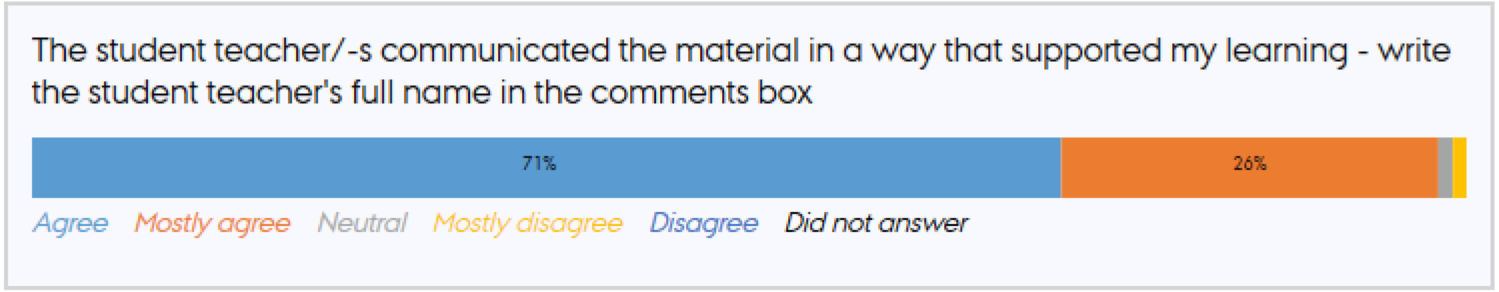






- Handin took often a lot of time
- Exercises hard to understand
- Recursion start was tough
- To many evaluation components (handins, project, exam)

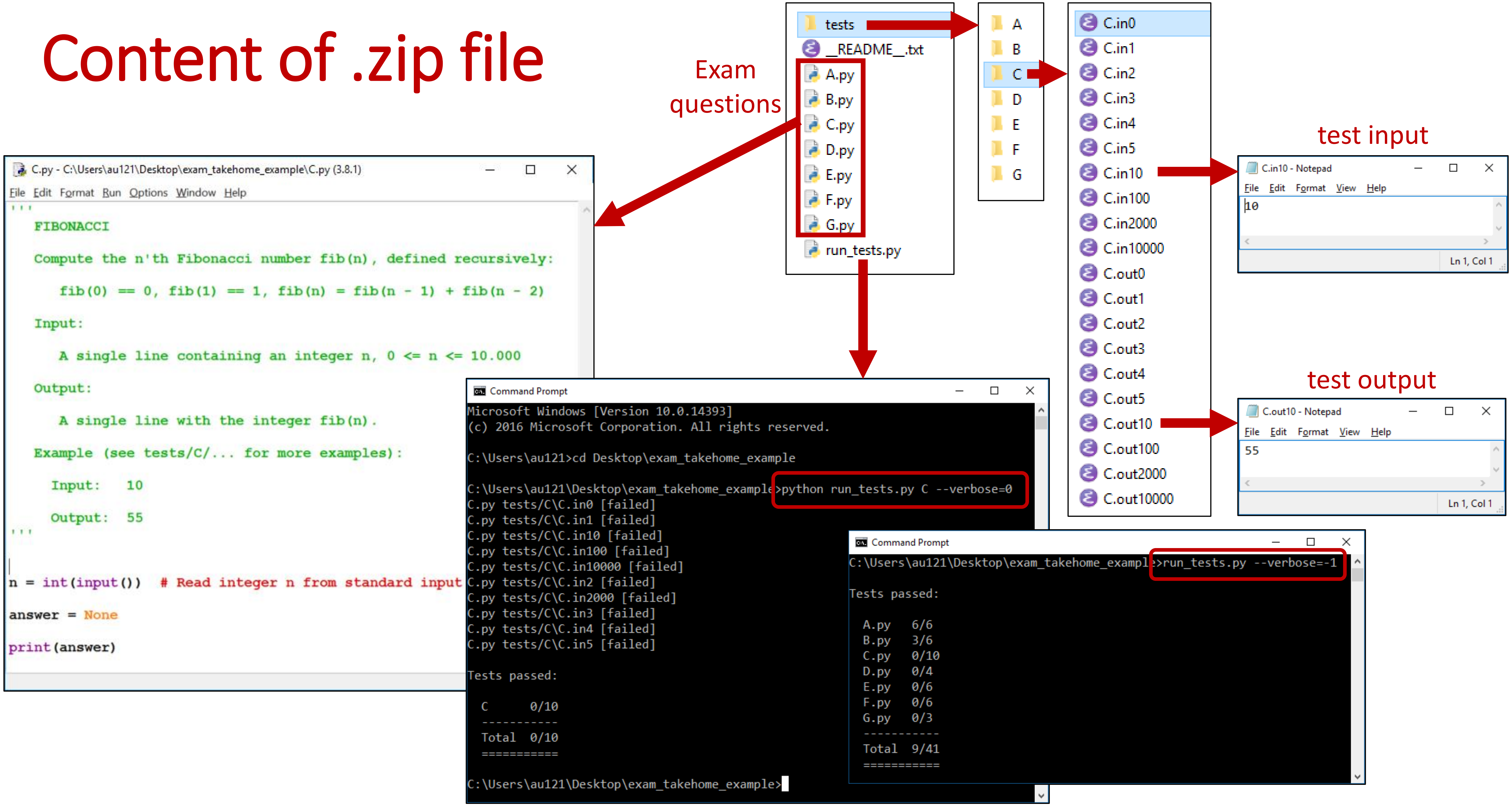
TAs - did an excellent job


The student teacher/-s communicated the material in a way that supported my learning - write the student teacher's full name in the comments box

Exam – 26 June 2020

- **6 hours, written exam, with aids, including PC and internet**
- **Communication with others about the exam is not permitted during the exam**
- Reexam in August (format likely the same, but no formal decision)
- Grade is an *overall assessment* of the implementation project and the exam
  - The result of the final exam must meet the minimum requirements for acceptance to be able to pass the course
  - The final exam will contribute roughly 3/4 to the final grade – but the final grade is an overall assessment
- eksamen.au.dk
  - Download .zip + add missing code + upload .zip
- Questions? – post them on Blackboard

Content of .zip file



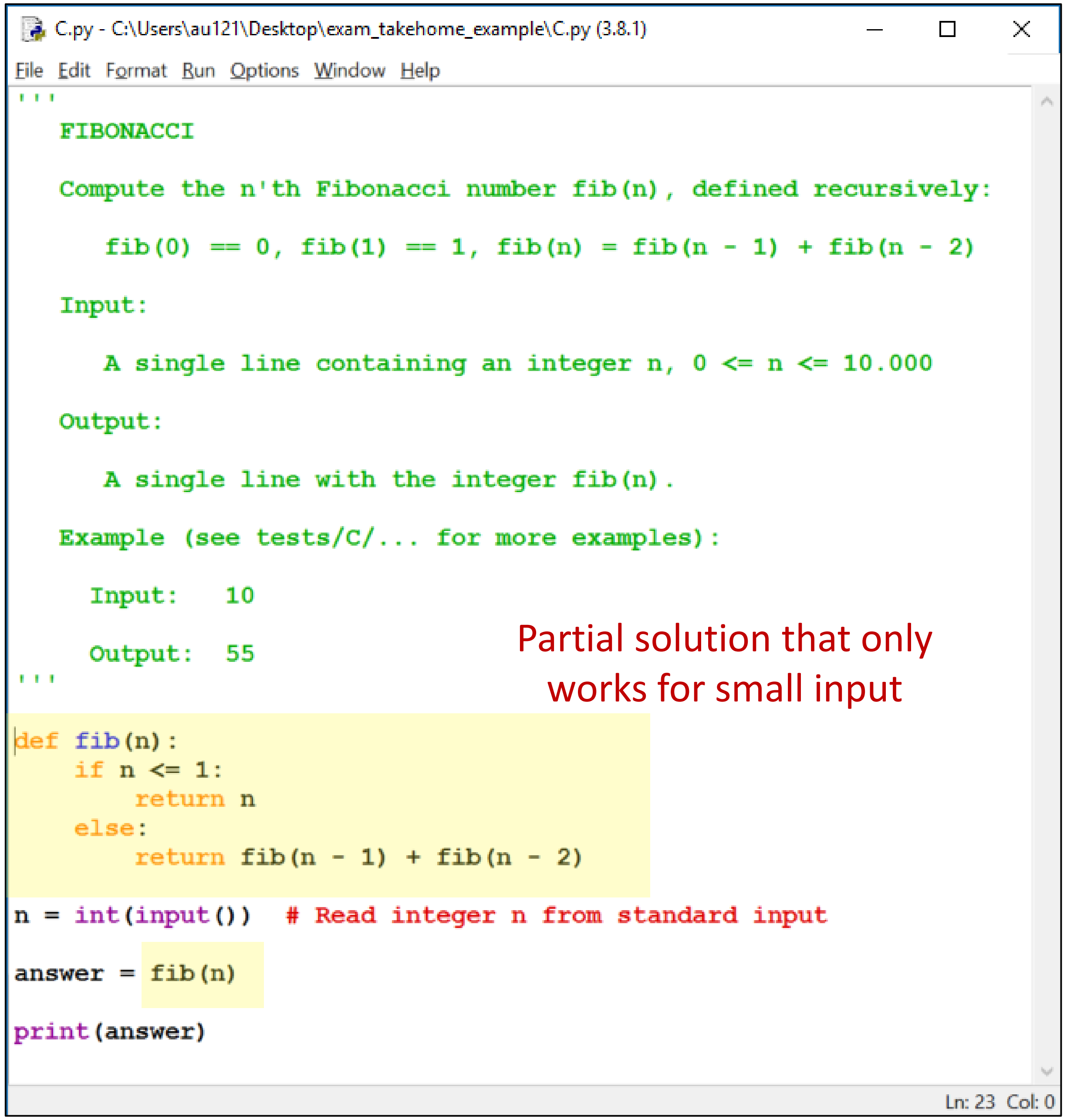
```
10
```

```
55
```

```
'''
   FIBONACCI

   Compute the n'th Fibonacci number fib(n), defined recursively:

      fib(0) == 0, fib(1) == 1, fib(n) = fib(n - 1) + fib(n - 2)

   Input:

      A single line containing an integer n, 0 <= n <= 10.000

   Output:

      A single line with the integer fib(n).

   Example (see tests/C/... for more examples):

     Input:   10

     Output:  55
'''

n = int(input())  # Read integer n from standard input

answer = None

print(answer)
```

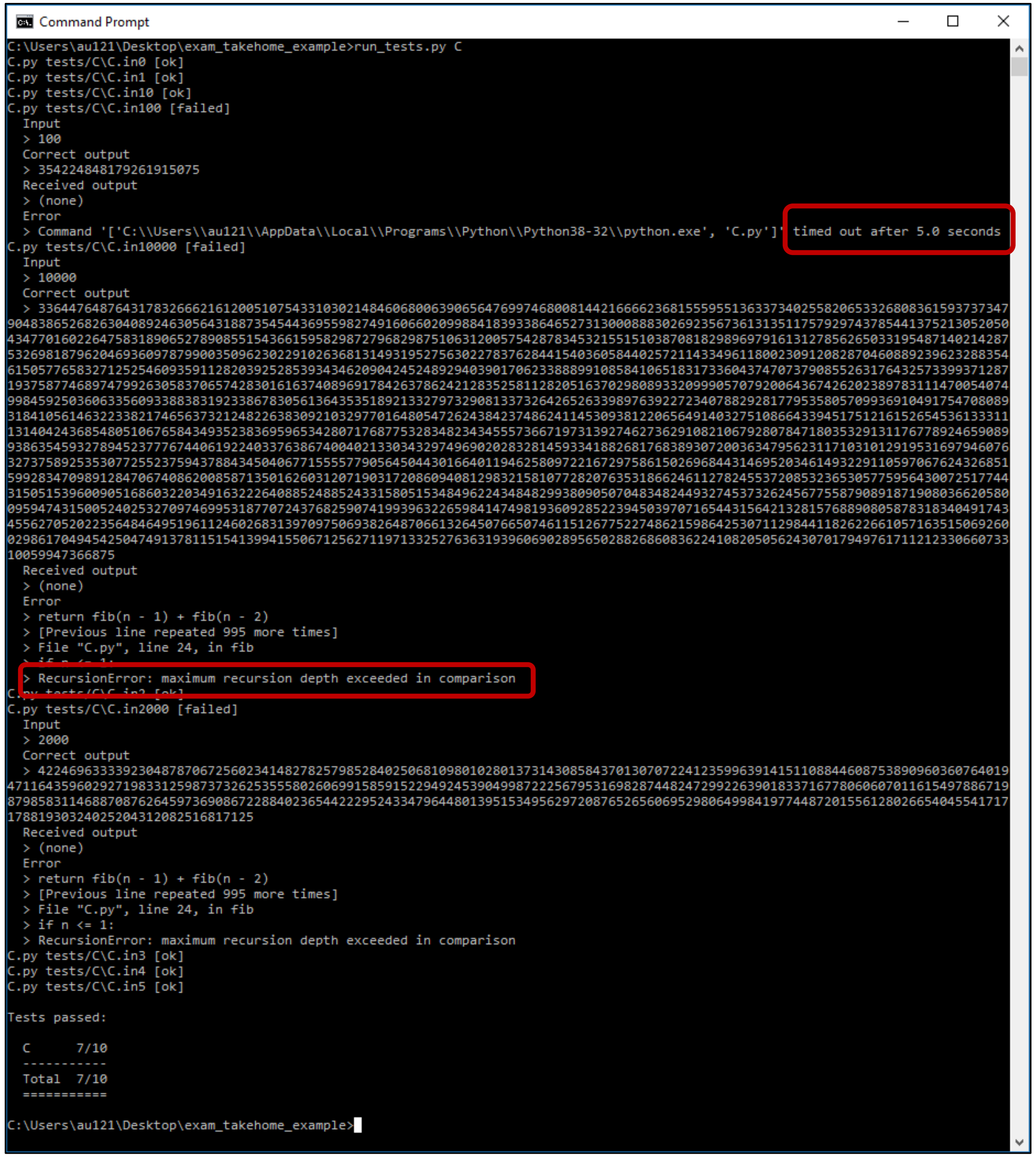
```
Microsoft Windows [Version 10.0.14393]
(c) 2016 Microsoft Corporation. All rights reserved.

C:\Users\au121>cd Desktop\exam_takehome_example

C:\Users\au121\Desktop\exam_takehome_example>python run_tests.py C --verbose=0
C.py tests/C\C.in0 [failed]
C.py tests/C\C.in1 [failed]
C.py tests/C\C.in10 [failed]
C.py tests/C\C.in100 [failed]
C.py tests/C\C.in10000 [failed]
C.py tests/C\C.in2 [failed]
C.py tests/C\C.in2000 [failed]
C.py tests/C\C.in3 [failed]
C.py tests/C\C.in4 [failed]
C.py tests/C\C.in5 [failed]

Tests passed:

  C      0/10
  ----------
  Total  0/10
  ==========

C:\Users\au121\Desktop\exam_takehome_example>
```

```
C:\Users\au121\Desktop\exam_takehome_exampl>run_tests.py --verbose=-1

Tests passed:

  A.py   6/6
  B.py   3/6
  C.py   0/10
  D.py   0/4
  E.py   0/6
  F.py   0/6
  G.py   0/3
  ----------
  Total  9/41
  ==========
```


C.py - C:\Users\au121\Desktop\exam_takehome_example\C.py (3.8.1)
File Edit Format Run Options Window Help
Ln: 23 Col: 0

```python
'''
   FIBONACCI

   Compute the n'th Fibonacci number fib(n), defined recursively:

      fib(0) == 0, fib(1) == 1, fib(n) = fib(n - 1) + fib(n - 2)

   Input:

      A single line containing an integer n, 0 <= n <= 10.000

   Output:

      A single line with the integer fib(n).

   Example (see tests/C/... for more examples):

     Input:   10

     Output:  55
'''

def fib(n):
    if n <= 1:
        return n
    else:
        return fib(n - 1) + fib(n - 2)

n = int(input())  # Read integer n from standard input

answer = fib(n)

print(answer)
```

Partial solution that only works for small input



```
C:\Users\au121\Desktop\exam_takehome_example>run_tests.py C
C.py tests/C\C.in0 [ok]
C.py tests/C\C.in1 [ok]
C.py tests/C\C.in10 [ok]
C.py tests/C\C.in100 [failed]
  Input
  > 100
  Correct output
  > 354224848179261915075
  Received output
  > (none)
  Error
  > Command '['C:\\Users\\au121\\AppData\\Local\\Programs\\Python\\Python38-32\\python.exe', 'C.py']' timed out after 5.0 seconds
C.py tests/C\C.in10000 [failed]
  Input
  > 10000
  Correct output
  > 33644764876431783266621612005107543310302148460680063906564769974680081442166662368155595513633734025582065332680836159373734790483865268263040892463056431887354544369559827491606602099884183933864652731300088830269235673613135117579297437854413752130520504347701602264758318906527890855154366159582987279682987510631200575428783453215515103870818298969791613127856265033195487140214287532698187962046936097879900350962302291026368131493195275630227837628441540360584402572114334961180023091208287046088923962328835461505776583271252546093591128203925285393434620904245248929403901706233888991085841065183173360437470737908552631764325733993712871937587746897479926305837065742830161637408969178426378624212835258112820516370298089332099905707920064367426202389783111470054074998459250360633560933883831923386783056136435351892133279732908133732642652633989763922723407882928177953580570993691049175470808931841056146322338217465637321248226383092103297701648054726243842374862411453093812206564914032751086643394517512161526545361333111314042436854805106765843493523836959653428071768775328348234345557366719731392746273629108210679280784718035329131176778924659089938635459327894523777674406192240337638674004021330343297496902028328145933418826817683893072003634795623117103101291953169794607632737589253530772552375943788434504067715555779056450443016640119462580972216729758615026968443146952034614932291105970676243268515992834709891284706740862008587135016260312071903172086094081298321581077282076353186624611278245537208532365305775956430072517744315051539600905168603220349163222640885248852433158051534849622434848299380905070483482449327453732624567755879089187190803662058009594743150052402532709746995318770724376825907419939632265984147498193609285223945039707165443156421328157688908058783183404917434556270520223564846495196112460268313970975069382648706613264507665074611512677522748621598642530711298441182622661057163515069260029861704945425047491378115154139941550671256271197133252763631939606902895650288268608362241082050562430701794976171121233066073310059947366875
  Received output
  > (none)
  Error
  > return fib(n - 1) + fib(n - 2)
  > [Previous line repeated 995 more times]
  > File "C.py", line 24, in fib
  > if n <= 1:
  > RecursionError: maximum recursion depth exceeded in comparison
C.py tests/C\C.in2 [ok]
C.py tests/C\C.in2000 [failed]
  Input
  > 2000
  Correct output
  > 4224696333392304878706725602341482782579852840250681098010280137314308584370130707224123599639141511088446087538909603607640194711643596029271983312598737326253555802606991585915229492453904998722256795316982874482472992263901833716778060607011615497886719879858311468870876264597369086722884023654422295243347964480139515349562972087652656069529806499841977448720155612802665404554171717881930324025204312082516817125
  Received output
  > (none)
  Error
  > return fib(n - 1) + fib(n - 2)
  > [Previous line repeated 995 more times]
  > File "C.py", line 24, in fib
  > if n <= 1:
  > RecursionError: maximum recursion depth exceeded in comparison
C.py tests/C\C.in3 [ok]
C.py tests/C\C.in4 [ok]
C.py tests/C\C.in5 [ok]

Tests passed:

  C      7/10
  ----------
  Total  7/10
  ==========

C:\Users\au121\Desktop\exam_takehome_example>
```

Evaluation of code

Don’t expect partial scores for this solution

```
def fib(n):
    if n == 10:
        return 55
    else:
        return None
```

- Each problem will be assigned a **weight**
- There will be problems of **varying difficulty**
  - import to be able to differentiate throughout
- Code will be evaluated on **known test cases** and **unknown test cases**
- In general **automatic scoring** in some exceptional cases manual

- Googling / stack overflow / Python documentation etc. **is allowed**, but put a **comment if you copied code from internet** to avoid plagiarism

AOB ?