Do not turn this page until you have received the signal to start.

(Please fill out the identification section above, write your name on the back of the test, and read the instructions below.)

Good Luck!

This midterm consists of 3 questions on 6 pages (including this one). When you receive the signal to start, please make sure that your copy is complete. Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can’t figure out how to write the code. No error checking is required: assume all user input and all argument values are valid.

If you use any space for rough work, indicate clearly what you want marked.

# 1: _____/ 4
# 2: _____/ 6
# 3: _____/ 10

TOTAL: _____/20
Question 1. [4 MARKS]

Part (a) [1 MARK] Write the output of the code below in the box below it.

```python
x = 3
y = x - 1
x = 4
print(x)
print(y)
```

Part (b) [1 MARK] Write the output of the code below in the box below it.

```python
s = 'coMPuTer'
i = len(s) - 1
while i > 0 and not s[i].isupper():
    print(s[i])
i = i - 1
```

Part (c) [1 MARK] Fill in the box with Python code that will make the program behaviour match the comments. You may not make any other changes to the code or add code outside the box.

```python
def within_capacity(num_passengers, within_limit):
    ''' (int, bool) -> bool
    Return True if and only if an airplane has fewer than 10 passengers
    and cargo weight is within the limit as indicated by within_limit.'''

    return [ ]
```

Part (d) [1 MARK] Fill in the box with Python code that will make the program behaviour match the comments. You may not make any other changes to the code or add code outside the box.

```python
s1 = "computer"
s2 = "science"

# Using only s1, s2, concatenation, and indexing and/or slicing, print the string 'test'.
print([ ])
```
Question 2. [6 marks]

Part (a) [4 marks]

For the function below, complete the function header (using a meaningful function name) and write a
docstring (including the type contract, description and two examples).

```python
def
    result = ''
    for ch in s:
        if ch.isupper():
            result = result + ch.lower()
        else:
            result = result + ch.upper()
    return result
```

Part (b) [2 marks]

Write a Python program that prompts the user to enter a string, calls the function from Part (a) passing
the string that the user entered as an argument, and prints the value returned by the function. You may
not change the function.
Question 3.  [10 MARKS]

Part (a)  [5 MARKS] Complete the function according to its docstring.

```python
def select_characters(s1, s2, selection):
    ''' (str, str, str) --> str

    Return a new string where each character is a character from either s1
    or s2 selected based on selection. selection is made up of 1s and 2s and
    indicates whether the character at the corresponding position of the new
    string should be from that position in s1 or that position in s2.
    s1, s2 and selection are the same lengths.

    >>> select_characters('coat', 'hard', '1221')
    'cart'
    >>> select_characters('pizza', 'hints', '11222')
    'pints'
    '''
```

```python
>>> select_characters('coat', 'hard', '1221')
'cart'

>>> select_characters('pizza', 'hints', '11222')
'pints'
```
Part (b) [5 MARKS] A library charges overdue fees for a borrowed book using the following fee schedule:

- less than 4 days late: 1 dollar per day
- 4 to 6 days late: 2 dollars per day (for all days, including the first 3 days)
- more than 6 days late: 3 dollars per day (for all days, including the first 6 days)

Borrowers of books are in one of these age groups: CHILD, ADULT or SENIOR. A CHILD gets charged only half of the fees and a SENIOR gets charged only one quarter of the fees. An ADULT pays the full fee.

Complete the following function according to the description above and the docstring below.

```
CHILD = 'child'
ADULT = 'adult'
SENIOR = 'senior'

def overdue_fees(days_late, age_group):
    ''' (int, str) -> number
    Return the fees for a book that is days_late days late for a borrower
    in the age group age_group.
    >>> overdue_fees(2, SENIOR)  # 2 days late, SENIOR borrower
    0.5
    >>> overdue_fees(5, ADULT)   # 5 days late, ADULT borrower
    10
    '''
```
Short Python function/method descriptions:

__builtins__:

- `input([prompt]) -> str`
  
  Read a string from standard input. The trailing newline is stripped. The prompt string, if given, is printed without a trailing newline before reading.

- `max(a, b, c, ...) -> value`
  
  With two or more arguments, return the largest argument.

- `min(a, b, c, ...) -> value`
  
  With two or more arguments, return the smallest argument.

- `print(value, ..., sep=' ', end='
') --> NoneType`
  
  Prints the values. Optional keyword arguments:
  - `sep`: string inserted between values, default a space.
  - `end`: string appended after the last value, default a newline.

- `int(x) -> int`
  
  Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero.

- `str`

  - `S.count(sub[, start[, end]]) -> int`
    
    Return the number of non-overlapping occurrences of substring sub in string S[start:end]. Optional arguments start and end are interpreted as in slice notation.

  - `S.find(sub[,i]) -> int`
    
    Return the lowest index in S (starting at S[i], if i is given) where the string sub is found or -1 if sub does not occur in S.

  - `S.isalpha() -> bool`
    
    Return True if and only if all characters in S are alphabetic and there is at least one character in S.

  - `S.isdigit() -> bool`
    
    Return True if and only if all characters in S are digits and there is at least one character in S.

  - `S.islower() -> bool`
    
    Return True if and only if all cased characters in S are lowercase and there is at least one cased character in S.

  - `S.isupper() -> bool`
    
    Return True if and only if all cased characters in S are uppercase and there is at least one cased character in S.

  - `S.lower() -> str`
    
    Return a copy of S converted to lowercase.

  - `S.replace(old, new) -> str`
    
    Return a copy of string S with all occurrences of the string old replaced with the string new.

  - `S.split([sep]) -> list of str`
    
    Return a list of the words in S, using string sep as the separator and any whitespace string if sep is not specified.

  - `S.startswith(prefix) -> bool`
    
    Return True if S starts with the specified prefix and False otherwise.

  - `S.strip() -> str`
    
    Return a copy of S with leading and trailing whitespace removed.

  - `S.upper() -> str`
    
    Return a copy of S converted to uppercase.