Temporary title page *Temporary title page* *Temporary title page*

CSSE1001/7030

2024 Sem1

MIDSEM

Temporary title page *Temporary title page* *Temporary title page*

§Multiple Choice

The following 10 questions are worth 1 mark each for a total of 10 marks.

Very important instruction: `Error` is the correct answer for any question with code that throws an error of any kind.

Question 1. [1 MARK]

What is the appropriate type-hint for the following function, assuming the function contains valid code and can be called without generating an error?

```
def foo(data, key):
    acc = 0
    for k in data[key]:
        acc += k
    return acc // len(data[key])
```

A. `foo(data: dict[int, list[int]], key: str) -> float`

[✓ B. `foo(data: dict[str, list[int]], key: str) -> int`]

C. `foo(data: dict[int, list[str]], key: str) -> float`

D. `foo(data: dict[str, list[str]], key: str) -> int`

E. More than one of the above.

Question 2. [1 MARK]

What is the value of `y` after *only* the following has been executed.

```
y = 0
for x in "ABCDE":
    if x == 'B' or 'C' or 'D' or 'E':
    ^   y += 1
    ^
```

This is the same as
if x == 'B' or True or True or True

A. `0`

B. `4`

[✓ C. `5`]

D. Error

E. None of the above.

Question 3. [1 MARK]

What is the value of xs after *only* the following code has been executed?

```
ys = [0, 1]
xs = [ys, ys]
ys.extend([2])
```

A. [[0, 1], [0, 1]]

B. [[0, 1, 2], [0, 1]]

[✓ C. [[0, 1, 2], [0, 1, 2]]]

D. Error

E. None of the above.

Question 4. [1 MARK]

What is the best description of the behaviour of the following function?

```
def bar(xs: list) -> bool:
    i = -1
    for x in xs:
        i += 1
    return xs and xs[i] != x:
```

^ typo.

So we accepted two solutions

A. bar checks if all members of xs are *the same*.

B. bar checks if all members of xs are *different*.

C. bar *always* returns True.

[✓ D. bar *always* returns False.]

[✓ E. bar *always* throws errors.]

Question 5. [1 MARK]

What is the value of x after *only* the following code is run?

```
x = 1

def foo(y) -> int:
    global x
    x = x + y
    return

foo(x)
foo(x)
```

A. 1

B. 2

[✓ C. 4]

D. Error.

E. None of the above.

Question 6. [1 MARK]

What is the value of `xs` after the following is evaluated?

```
xs = "toque"
ys = xs[::]
ys[0] = "T"
```

Remember: Strings are immutable

A. `""` (empty string)

B. `"toque"`

C. `"Toque"`

[✓ D. `Error`]

E. None of the above.

(Author's note: "toque" is the Canadian word for "beanie").

Question 7. [1 MARK]

Which of the following expressions *can* generate an error when `x = dict()`.

A. `x[(1,)] = 1`

B. `x["1"] = 1`

[✓ C. `x[1] == 1`]

Only option that is taking something OUT of a dict (rather than putting something in)

D. `1 in x`

E. More than one of the above.

Question 8. [1 MARK]

What is the value of `x` after running the following code?

```
s = 'drake is mid'
x = s[:-3:-1]
```

[✓ A. `'di'`]

B. `'id'`

C. `'dim'`

D. `'mid'`

E. `''` (empty string)

Question 9. [1 MARK]

How many of the following statements evaluate to `True`?

```
>>> bool("False")
>>> bool(True > False)
>>> bool([1, 2, 3] and 1/0)
                    ^
```

We meant "or" here.

We accept two solutions as a result.

A. 0

B. 1

[✓ C. 2]

D. 3

[✓ E. Error]

Question 10. [1 MARK]

Consider the function `foo` defined below that computes the salary of a minimum wage employee that has worked `hours` number of hours. What *type* of value does `foo` return?

```
def foo(hours: int):
    """
    Precondition:  hours > 0
    """
    salary = 23.23 * hours
    return None  # default behaviour
```

A. `int`

B. `float`

C. `str`

D. `char`

[✓ E. None of the above.]

§Very Short Answer

Write your solutions inside the box. *All writing outside boxes will be ignored.*

The following *three* questions are worth a total of 5 marks.

Question 11. [1 MARK]

What does the following arithmetic expressions evaluate to?

```
(1 + 1) ** 3 % 8 - 2 ** 1 ** 2 - 20 // 4
```

-7

Question 12. [2 MARKS]

Re-write the following code snippet so that it uses *while loops* instead of *for loops*. Assume that `z` is of the type `list[list[str]]`.

```
ans = 0
for y in z:
    for x in y:
        ans += len(x)
```

```
ans, i = 0, 0
while i < length(z):
    j = 0
    while j < length(z[i]):
        ans += len(z[i][j])
        j += 1
    i += 1
```

Question 13. [2 MARKS]

Suppose we have run the following code and the user has typed something.

```
value = input("Enter a single digit: ")
```

Write a Python expression that evaluates to `True` *only when* the user has typed a *single digit* (and `False` otherwise).

```
value in "0123456789" and len(value) == 1
```

§Very Long Answer

The following two questions are worth 5 marks each for a total of 10 marks.

Question 14. [5 MARKS]

Implement the following function according to the specification. *Do not* include a docstring.

You are *not* permitted to use the string method `.title()` for this question; solutions that use this method will receive zero marks.

```
def foo(xs: str) -> str:
    """
    Return the input string <xs> converted to title-case.

    This is, return a version of the input spring where words start with
    uppercased characters and all remaining cased characters have lower case.

    >>> foo("run spot run")
    'Run Spot Run'
    >>> foo("rUn SpOt rUn")
    'Run Spot Run'
    >>> foo("a b 1 %")
    'A B 1 %'
    """
```

The following list methods may be useful here:

```
>>> help(str.lower)
lower(self, /)
    Return a copy of the string converted to lowercase.

>>> help(str.upper)
upper(self, /)
    Return a copy of the string converted to uppercase.
```

Write your answer on the next page.

Write your answer on the next page.

Write your answer on the next page.

Question 14 continued. . .

```
def foo(xs: str) -> str:

    ans = " "
    for x in xs:
        if ans[-1] == " ":
            ans += x.upper()
        else:
            ans += x.lower()

    return ans[1:]
```

Question 15. [5 MARKS]

Implement the following function according to the specification. *Do not* include a docstring.

```
def foo(xs: str) -> str:
    """ Return the character of <xs> that occurs most frequently.

    If there is a tie for the most frequent element the element with least
    index is returned.

    Precondition:
       len(xs) > 0

    >>> foo("AAABB")
    'A'
    >>> foo("AABBB")
    'B'
    >>> foo("ABABAB")
    'A'
    >>> foo("BABABA")
    'B'
    """
```

Write your answer on the next page.

Write your answer on the next page.

Write your answer on the next page.

Question 15 continued…

```
# either define this or use str.count
def count(y, ys: str) -> int:
    ans = 0
    for x in ys:
        if y == x:
            ans += 1
    return ans


def foo(xs: str) -> str:
    cur_ans, cur_freq = '', -1

    for x in xs:
        if count(x, xs) > cur_freq:
            cur_freq = count(x, xs)
            cur_ans = x

    return cur_ans
```

END OF EXAMINATION