Midterm Exam (Spring 2025)

Instructions

Please write your name and university-issued email address below in the space provided.

Name: _____

Email Address: _____

You will have 75 minutes to answer the questions contained herein. You may submit the exam at any time within that period. Once you begin the exam, you may not leave the room until you submit it.

You are expected to not consult with any other source of information during the exam period. No phones! Please take a moment before the exam starts to clear your desk of all materials except for your cheat sheet. There should be no talking for any reason during the exam period. If you have a question about the exam material, raise your hand and wait for an opportunity to ask an instructor for clarification.

When you are ready, you may begin. Good luck!

Evaluation

The weight of each section is detailed below. Partial credit may be awarded, and there is no penalty for guessing, so you are encouraged to not leave any questions blank.

The exam will be graded "on a curve". Keep focused, stay light, and don't spend too long on any single question. If you get stuck on one question, you are encouraged to move on and pick up as many points as possible.

Exam Part	**Weight**
Part I: Computer Science Concepts	18%
Part II: Python Datatypes	12%
Part III: Code Tracing	18%
Part IV: Control Flow	16%
Part V: Data Processing	
… A. Captions	12%
… B. Speaker Names	12%
… C. Speaker Analysis	12%

I. Computer Science Concepts [18%]

1. What is the role, or purpose, of **software** within the context of a computer-based information system? What is the role, or purpose, of **hardware**? And what is the relationship between software and hardware in such a system?

2. When we write computer programs, we have a choice of which language to program in. What are three distinct reasons **why Python** is a good choice? In other words, what are some positive aspects or benefits of programming in the Python language (in general, and/or as opposed to other languages)?

    a.

    b.

    c.

3. Computer programs are limited by multiple factors, including **disk space** and **memory**. For each of the following real life scenarios, identify whether it primarily uses disk space or memory. Answer with either "Disk Space" or "Memory".

    a. You download a movie onto your tablet to watch later. *Answer*: _____

    b. You have a lot of applications open on your computer. *Answer*: _____

    c. You copy a large video file to an external hard drive. *Answer*: _____

    d. You open a large spreadsheet containing millions of rows. *Answer*: _____

    e. Your Python program stores a value in a variable. *Answer*: _____

    f. Your Python program writes a text file to the filesystem. *Answer*: _____

II. Python Datatypes [12%]

4. For each of the following example Python objects, what is its **datatype**? FYI: for nested objects, specify only the datatype of the parent (outermost) object. Provide your answers in the column on the right.

   a. `True` *Datatype*: _____

   b. `None` *Datatype*: _____

   c. `99` *Datatype*: _____

   d. `9.99` *Datatype*: _____

   e. `"Buy our products!!"` *Datatype*: _____

   f. `["AAPL", "MSFT", "GOOGL", "NFLX", "SPOT"]` *Datatype*: _____

   g. `{"symbol": "AAPL", "product": "iPhone"}` *Datatype*: _____

   h. `[{"symbol": "AAPL"}, {"symbol": "GOOGL"}]` *Datatype*: _____

   i. `[85, 90, 95, 100]` *Datatype*: _____

   j. `"AAPL"` *Datatype*: _____

   k. `{"symbol": "AAPL", "sales": [85, 90, 95, 100]}` *Datatype*: _____

   Reference the provided `message` variable. What is its **datatype**?

   ```
   message = ("HELLO" + " " + "WORLD") * 6
   ```

   l. `message` *Datatype*: _____

   Reference the provided `result` variable. What is its **datatype**?

   ```
   result = ({"symbol": "AAPL"} == {"symbol": "GOOGL"})
   ```

   m. `result` *Datatype*: _____

III. Code Tracing [18%]

5. Reference the code provided below. There are a number of **print statements**. For each print statement, **what result will we see**? Provide your answers in the "output" space below each corresponding "input". If there are multiple print statements, make sure to display them all. Case matters, so ensure the results you write observe the proper capitalization.

a. Input:

```
if 5 + 5 == 10:
    print("YEP")
print("BOTTOM")
```

Output:

b. Input:

```
if 5 + 5 != 10:
    print("YEP")
print("BOTTOM")
```

Output:

c. Input:

```
letters = list("abcdef")
print(letters)
```

Output:

d. Input:

```
message = "good luck. have fun!!!"

message = message.replace(".", "").replace("!", "")
message = message[4:]
message = message.upper()
message = message.split(" ")

print(message[-1])
```

Output:

e. Given:

```
def calculate_bonus(sales):
    if sales >= 1000:
        rate = 0.10
    else:
        rate = 0.05

    return sales * rate

# UNIT TESTS (ASSUME ALL ARE PASSING):
assert calculate_bonus(600) == 30.0
assert calculate_bonus(800) == 40.0
assert calculate_bonus(1000) == 100.0
assert calculate_bonus(1200) == 120.0
```

Input:

```
total_bonus = 0

employees = {'AJ': 1100, 'Dan': 700, 'Priya': 1500}

for employee, sales in employees.items():
    bonus = calculate_bonus(sales)

    print(employee.upper(), "BONUS:", bonus)

    total_bonus += bonus

print("------------")
print("TOTAL BONUS:", total_bonus)
```

Output:

IV. Control Flow [16%]

Provide the **Python code** needed to run each of the simulations below. NOTE: you do not need to provide the numeric solution (no mental math or calculators necessary)! Just provide the Python code needed to arrive at the solution!

6. High Yield Savings Simulation:

   You have saved $150,000 in the bank. You decide to transfer this money into a high yield savings account. The rate of return for this savings account is 5.5% per year. Each year you transfer an additional $20,000 into the savings account. NOTE: At the end of each year, the annual return is calculated and deposited into your account AFTER this additional contribution has been incorporated into the balance of the account. Under these conditions, how many full years will it take your money to grow to $1,000,000? Display / print the **number of years** (i.e. `18`).

7. Retirement Longevity Simulation:

   You have saved $2,500,000 in the bank. You decide to retire! Last year, each month you spent $3,500 on rent, and $4,500 on your credit card bill. This year, and each year into the future, your monthly rent is projected to rise by 10% per year, and your monthly credit card bill is projected to rise by 2.5% per year. NOTE: to calculate the annual expenses for a given year, you can multiply the monthly expenses by 12 AFTER incorporating the increases. Under these conditions, how many full years will it take for your money to run out? Display / print the **number of years** (i.e. `16`).

V. Data Processing (Crunch the Zoom Transcript Data)

See the Python variable called `captions` provided at the end of the exam booklet. This data represents a transcript from a Zoom video call meeting between teammates at a software development company. Use this data to answer all the questions in this section.

NOTE: in this `captions` data, assume the speaker name appears at the beginning of the caption text, and is followed by a colon (":") to separate the speaker name from what they said (i.e "speaker name: what they said"). For simplicity, assume there are no other colons present in the caption text.

A. Captions [12%]

8. Given the provided `captions` variable, **write Python code** to answer the following questions.

    a. Display / print the number of captions (i.e. `11`).

    b. Display / print the full text of the first caption (i.e. `'Hiroshi: Good morning, everyone. Let's get started. Today we're discussing the next steps for the product design and development.'`).

    c. Display / print the total duration of this meeting, which is the same as the end time of the last caption (i.e. five minutes, or `'00:05:00'`). NOTE: no math or calculations necessary.

    d. Loop through the captions, and print the start time and full text of each.

B. Speaker Names [12%]

9. Given the provided `captions` variable, **write Python code** to answer the following questions.

    a. Display / print the name of the first speaker (i.e. `'Hiroshi'`). Remember, you can assume the caption text is in the format of "speaker name: what they said").

    b. Write a custom function called `parse_speaker_name` to output the speaker name for any given caption text. The function should accept a parameter input called `caption_text` (the caption text, assumed to be in the format of "speaker name: what they said"). The function should return the corresponding speaker name.

    The working version of this function should make the following unit tests pass:

    ```
    example_text = "Hiroshi: Good morning, everyone. Let's get started..."
    assert parse_speaker_name(example_text) == "Hiroshi"

    assert parse_speaker_name("Anna: I love coding!") == "Anna"
    ```

    NOTE: regardless of whether you get this question correct, in future questions, you can reference / use the `parse_speaker_name` function, as if it is functioning properly, and passing the tests.

    c. Use the `parse_speaker_name` function to display / print the name of the last speaker.

C. Speaker Analysis [12%]

10. Given the provided `captions` variable, **write Python code** to answer the following questions.

    a. Who spoke? Collect a list of unique speaker names (i.e. `['Hiroshi', 'Aisha', 'Carlos']`) and store them in a variable called `speaker_names`. Then print this list of speakers.

       NOTE: regardless of whether you get this question correct, in future questions, you can reference / use the `speaker_names` variable, as if it has the correct values.

    b. How many times did each speaker speak? For each unique speaker, print their name, and the number of times they spoke in total. You can either print the results for each speaker separately (each on its own line), or as a single dictionary (i.e. `{'Hiroshi': 5, 'Aisha': 4, 'Carlos': 2}`).

This page has been left intentionally blank. Feel free to make notes on it. Its contents will not be evaluated.

---

*The `captions` variable provided on the following two pages is to be used in conjunction with in the **Data Processing** part of the exam. Feel free to detach these pages and use them as a visual reference during the exam. If you do detach them, write your name on them and remember to return them at the end of the exam along with the rest of your exam booklet!*

```
captions = [
  {
    'id': 1,
    'start': '00:00:00',
    'end': '00:00:20',
    'text': "Hiroshi: Good morning, everyone. Let's get started. Today
we're discussing the next steps for the product design and development."
  },
  {
    'id': 2,
    'start': '00:00:20',
    'end': '00:00:45',
    'text': 'Aisha: Right, so we have the initial prototype ready. The user
feedback we collected has been positive, but there are a few concerns about
the usability.'
  },
  {
    'id': 3,
    'start': '00:00:45',
    'end': '00:01:10',
    'text': 'Hiroshi: Agreed. We need to address the navigation flow and
some interface issues. Our goal is to simplify the user journey without
compromising on features.'
  },
  {
    'id': 4,
    'start': '00:01:10',
    'end': '00:01:40',
    'text': 'Carlos: I think we should also consider performance
optimization early on. The product needs to load faster, especially on
mobile devices.'
  },
  {
    'id': 5,
    'start': '00:01:40',
    'end': '00:02:05',
    'text': 'Aisha: Absolutely, mobile optimization is a priority. We
should start looking at integrating some responsive design techniques in
this phase.'
  },

  # ... (continued on next page)
```

```
  # ... (continued from previous page)

  {
    'id': 6,
    'start': '00:02:05',
    'end': '00:02:35',
    'text': "Hiroshi: Let's also think about the color scheme and
typography. We need something modern but still professional to appeal to
our target audience."
  },
  {
    'id': 7,
    'start': '00:02:35',
    'end': '00:03:00',
    'text': "Aisha: I'll have the design team draft a couple of options by
the end of the week. Does everyone agree to review them next Monday?"
  },
  {
    'id': 8,
    'start': '00:03:00',
    'end': '00:03:30',
    'text': "Carlos: That works for me. In the meantime, I'll start working
on improving load times and checking compatibility with older browsers."
  },
  {
    'id': 9,
    'start': '00:03:30',
    'end': '00:04:00',
    'text': "Hiroshi: Sounds good. Let's set another meeting for next
Friday to finalize everything before our presentation to stakeholders."
  },
  {
    'id': 10,
    'start': '00:04:00',
    'end': '00:04:30',
    'text': "Aisha: Great! I think we're on track, but let's keep
communication open if anyone runs into any issues."
  },
  {
    'id': 11,
    'start': '00:04:30',
    'end': '00:05:00',
    'text': "Hiroshi: Thanks, everyone! Let's make sure we're hitting our
milestones. Meeting adjourned."
  }
]
```