

## i

**Self-declaration**

I hereby declare that the submission I am submitting is my own work.

**I have not:**

- collaborated with other students
- used the work of others without stating this
- used my own previous work (submissions/exam answers) without stating this

If I have used literature, a bibliography will contain all the sources I have used in the assignment and references will refer to this list.

**I am aware that violation of these provisions is considered cheating and may lead to cancellation of the exam and/or exclusion.**

If you are unsure whether you can support this declaration, see the guidelines for the use of sources in written work at the University of Bergen, and contact your study counsellor/course coordinator.

All exam papers at UiB are sent for manual and electronic plagiarism control.

**Please note: By continuing, I confirm that I have read the declaration and that the paper I am submitting for this exam is my own work (and only my own work), in full compliance with the above declaration.**

## i

**Exam Information**

The exam consists of three parts, totalling 80 points:

Type	Automatically graded questions	Explanation questions	Coding questions
Points	46	16	18

- Part 1 consists of test questions arranged in random order. In case of difficulty, you can skip any question and return to it later. To avoid forgetting skipped questions, we recommend using the flagging option in the upper right corner.
- You should have started parts 2 and 3 when you have 2 hours left. In part 3, you will not have the opportunity to run the code you write. The examiner is aware of this and will not deduct points for minor errors such as typos in function names from the standard library. However, you must write the code as clearly and correctly as possible to demonstrate that you understand the nuances of the code you write.
- Permitted aids for the exam: all written and printed aids.

**General advice**

- Read the question before you answer
- Work quickly through all the questions in the first round, and come back to challenging questions again if you have time at the end.
- Help the sensor to help you! If you are unsure about the interpretation of a question, make a brief comment on how you interpret uncertainties in the question. If you can't remember exactly how to do something with code, write a comment explaining what you're trying to do.

## i

**Materials**

On this page, you will find all the course notes in the course collected in a PDF. They are available to you if you wish. It is completely voluntary to use them. Normal citation rules apply: if you copy something from the course notes, it is important that you cite them as a source in your answer.

You can also find the course notes in the bottom left corner throughout the exam. Then the notes will open in a new tab.

PS: Have the pages suddenly turned sideways? Try pressing 'R' to turn them the right way round again.

1(a)

```

pump_A = 5
pump_B = 2
pump_A = pump_A + pump_B
pump_B = pump_A * pump_B
pump_A -= 1
print(pump_B - pump_A)

```

**Gas station** have two petrol columns: `pump_A` and `pump_B`. Noah noticed that the programme that was supposed to calculate the amount of gas left in the pump started to go crazy and perform completely counterintuitive operations. He decided to test it with prints: What does this program print? (if the program crashes, just print Error)

Fill in your answer  here.

---

Maximum marks: 2

1(b)

```

brann_squad = {
    "forwards": [
        {"name": "Niklas Castro",
         "career": ["Vålerenga", "Kongsvinger", "Aalesunds", "Brann"],
         "physical": {
             "age": 29,
             "foot": "right",
             "height": 173
         }
        },
        {"name": "Aune Selland Heggebø",
         "career": ["Brann"],
         "physical": {
             "age": 23,
             "foot": "left",
             "height": 185
         }
        }
    ]
}

```

Olav decided to create a database of **SK Brann footballers** for his informatics project. He decided to store players in a dictionary format. Below, you can see an example of first two observations:

Now he wants to test whether his structure works correctly.

Answer the questions below (if the program would crash on the line, just write Error):

**print Niklas Castro's age (integer):**

- ☐ `print(brann_squad[0][2][0])`
- ☐ `print(brann_squad["Niklas Castro"]["age"])`
- ☐ `print(brann_squad["Niklas Castro"]["physical"]["age"])`
- ☐ `print(brann_squad["forwards"]["Niklas Castro"]["age"])`
- ☐ `print(brann_squad["forwards"][0]["physical"][0])`
- ☐ `print(brann_squad["forwards"][0]["physical"]["age"])`

**print Aune Heggebø's first club (string)**

- ☐ `print(brann_squad[1][0][0])`
- ☐ `print(brann_squad["forwards"][0]["career"][0])`
- ☐ `print(brann_squad["forwards"][1]["career"][0])`
- ☐ `print(brann_squad["forwards"][2]["career"][0])`
- ☐ `print(brann_squad["forwards"][2]["career"][1])`
- ☐ `print(brann_squad["forwards"]["Aune Selland Heggebø"]["career"][0])`

---

Maximum marks: 2

1(c)

```
def is_vaccine_distributable(status_by_country):
    lead_approval = status_by_country[0]

    if lead_approval == 0:
        # no approval from lead research country
        return False

    for outbreak in status_by_country[1:]:
        if outbreak == 1:
            # active outbreak in a partner country
            print(status_by_country.index(outbreak), "fails the test")
            return False

    # all good :)
    return True

is_vaccine_distributable([1, 0, 0, 1, 0])
```

Ingrid is a lead analyst in health regulator company. Before **distributing a vaccine** globally, she needs to make sure it meets 2 conditions:

1. The lead research country must have approved the vaccine.
2. None of the partner countries should currently have a critical outbreak.

The lead research country must have approved the vaccine.

None of the partner countries should currently have a critical outbreak.

She wrote the `is_vaccine_distributable()` function that checks these conditions:

However, the data about countries is messed. In the input list:

- the first value is the lead country approve status,
- followed by the partner countries outbreak status.

What will Ingrid's function return with `[1, 0, 0, 1, 0]` input?

Fill in your answer here:



---

Maximum marks: 2

1(d)

```
def medicine_reminder(days):
    for day in range(1, days + 1):
        if day % 3 == 0:
            print(f'{day} drink your medicine!')
        else:
            print(day)
```

Ida is interning at a medical center where her job is to automate reminders based on **healthcare data** of the patients. For example, she wrote the function below:

What would be the last day when this function will print *"drink your medicine!"* in February 2025 (contains 28 days)? Write an integer below (if the programme crashes, write only Error).

Answer:



---

Maximum marks: 2

**1(e)** The **World Health Organization (WHO)** uses a simple system to flag individuals who may need further follow-up in a vaccination campaign. Each person is assessed based on:

- Whether they've had contact with ill person recently (variable name `contact`)
- Whether they've had symptoms of the disease (`symptoms`)
- Whether they're vaccinated (`vaccinated`)

Each individual is flagged for follow-up based on two criteria:

1. The person that contacted with ill person have symptoms.
2. The person who hasn't been vaccinated either contacted an ill person, or have symptoms.

Write down both conditions below as logical expressions in Python, using 3 variables from above:

flag\_1 =

flag\_2 =

Maximum marks: 2

**1(f)** Martha is **knitting** a cozy winter sweater as a gift for her cousin. She's know that it will need 50000 stitches, and now she's choosing between several types of yarn - Wool, Cotton, and luxurious Cashmere - and wants to know how many balls of yarn (clews) she'll need. Each type of yarn allows a different number of stitches per ball (7000, 9000 and 12000 respectively).

Help Martha figure out how many balls of yarn she'll need to complete a full sweater; write a function that takes yarn type as an argument and returns an integer amount of balls that will be enough for this sweater.

**Complete the function below**

Test case #	Input	Expected output
1	<code>is_enough_yarn("Wool")</code>	8
2	<code>is_enough_yarn("Cashmere")</code>	5
3	<code>is_enough_yarn("Cotton")</code>	6

```
def is_enough_yarn(yarn_type):
    # Estimated total stitches needed for one sweater
    stitches_per_sweater = 50000

    # Estimated stitches per clew, based on yarn_type (from description)
    ...
    ...
    ...

    # Calculate number of clews needed
    clews_needed = ...
    return clews_needed
# DO NOT EDIT THE LINE BELOW, OTHERWISE THE AUTOTEST WILL NOT WORK
print(eval(input()))
```

Test code

Maximum marks: 6

- 1(g) Emil decided to try yourself in the exciting sports of **disc (or frisbee) golf**, but he's not sure which discs should he pick for each hole. He found an algorithm on the internet, but some functions were blurred for some reason. Your task is to fill the gaps in the function with proper keywords or operators, so Emil can decide which disc to choose:

Match a correct answer with the blank space in the code:

```
_____ pick_disc(distance, wind, angle):
```

```
    if distance > 100:
        if wind < 10 _____ angle > 50:
```

```
            disc = "driver"
```

```
        else:
```

```
            disc = "midrange"
```

```
    _____ distance > 50:
```

```
        disc = "midrange"
```

```
    else:
```

```
        disc = "putter"
```

```
    _____ disc
```

### Alternatives:

elif

def

return

and

Maximum marks: 2

- 1(h) Marius is setting up the system for **files navigation at his PC** like a pro. He needs to develop a function that takes the name of a file as an argument, checks if the filename ends with `.txt`, `.png`, or other extensions, and returns a string message about the file type.

Help Marius finish this function system by filling the gaps below.

Complete the function below

Test case #	Input	Expected output
1	get_file_format("IMG3906.png")	This is an image
2	get_file_format("script.py")	Unknown or executable file
3	get_file_format("README.txt")	This is a text file

```
def get_file_format(filename):
    ...
    # DO NOT EDIT THE LINE BELOW, OTHERWISE THE AUTOTEST WILL NOT WORK
    print(eval(input()))
```

Test code

Maximum marks: 6

- 1(i) Imagine that, inspired by Emily Dickinson (whose work is often succinct and rhythmically compact), Maja is working on a tool that filters **poetry** lines. The function filters lines to keep only those shorter than 32 characters - ideal for old tweet-sized fragments.

However, several parts of the code were smudged by ink. Now Maya needs to fill in the missing Python syntax, so the program works as intended.

Match a correct answer with the blank space in the code.

```
poem_lines = [
    "“Hope” is the thing with feathers",
    "That perches in the soul",
    "And sings the tune without the words",
    "And never stops - at all"
]
```

```
def select_lines(poem_lines):
    selected = []
    ----- line ----- poem_lines:
    ----- len(line) < 32:
    ----- selected.append(line)
    ----- selected

print(select_lines(poem_lines))
```

Alternatives:

if

return

for

in

Maximum marks: 2

1(j)

```
def check_outage(voltage_data, limit):
    for district in voltage_data[1:]:
        if district[1] < limit:
            district[2] = 'Outage'
voltage_data = [
    ['district', 'is_voltage', 'status'],
    ['Bønes', 120, 'OK'],
    ['Varden', 0, 'OK'],
    ['Sædalen', 350, 'OK'],
    ['Fantoft', 0, 'OK'],
    ['Åsane', 700, 'OK'],
]

new_voltage_status = check_outage(voltage_data, 10)
```

William is working with a team that monitors **electricity supply** in different districts of Bergen. Each district is checked for voltage levels, and if the voltage drops below a critical limit, the district is marked as experiencing an outage.

He wrote the function `check_outage()` that takes in a list of voltage data and a limit value.

Help William with the testing of what this function does by *predicting the output of the following 4 expressions*:

Please match the values:

	list	NoneType	int	str	dict	ERROR
<code>print(type(voltage_data[3][3]))</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>print(type(voltage_data[1][1]))</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>print(type(new_voltage_status))</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
<code>print(type(voltage_data[2][2]))</code>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maximum marks: 2

1(k)

```
games_2004 = 'half-life 2, doom 3, grand theft auto: san andreas, metal gear solid 3: snake eater, world of warcraft'

big_games = games_2004.____()
print(big_games.split(', '))

# Output:
# ["HALF-LIFE 2", "DOOM 3", "GRAND THEFT AUTO: SAN ANDREAS",.. ]
```

Thomas is a passionate gamer. He found an example of **video games** that came out in 2004, which has been retrospectively considered one of the best and most influential in video game history. However, they are all written in lowercase letters, which is not very presentable. Thomas' task is to write a method uppercase all games in the string.

Fill the blank space in the code above.

Answer: games\_2004.  ()

---

Maximum marks: 2

1(l)

```
gallery_notes = '''
O'Keeffe: I had to create an equivalent for what I felt about what I was looking at—not copy it.
Kahlo: I paint flowers so they will not die.
O'Keeffe: I'm not a joiner and I'm not a precisionist or anything else.
Kahlo: Fall in love with yourself, with life and then with whoever you want.
O'Keeffe: My painting is what I have to give back to the world for what the world gives to me.
'''

quotes_by_line = gallery_notes.____()
kahlo_quotes = [quotes_by_line[1], quotes_by_line[3]]
print(kahlo_quotes)

# Output:
# ['Kahlo: I paint flowers so they will not die.',
# 'Kahlo: Fall in love with yourself, with life and then with whoever you want.']
```

The art gallery contains quotes from **famous painters**. In anticipation of the anniversary of Frida Kahlo, a famous artist in the genres of *primitivism* and *surrealism*, Aurora has been given the task of selecting her quotes, which are mixed with quotes by another modernist painter Georgia O'Keeffe. She wrote a script to elicit only certain lines from a converted list of quotes:

Write below which method is missing to help Aurora to run the code.

Answer: gallery\_notes.  ()

---

Maximum marks: 2

1(m)

```
def calculate_bonus(last_round_score):

    bonus = 0

    while last_round_score > 0:
        d = last_round_score % 10
        bonus += d
        last_round_score //= 10

    return bonus
```

Jacob is playing a fantasy **board game** with his friends. The game has an interesting new mechanic: each player gets a bonus at the end of the game based on their total score from the last round. To calculate the bonus, he wrote this function:

What will the function return, given that Jacob scored 49 points in the last round of the game? (if the programme crashes, write only Error).

Answer: .

---

Maximum marks: 2

- 1(n) **Barbie** is deciding whether to go on a beach trip. Her decision depends on several things: whether Sasha is available, whether Gloria is coming, whether the sunset pretty, and whether there are good snacks. If all that fails, she might go with Ken anyway (but only if it's not close to a nighttime: because every night is girls' night).

This is the logic behind her thinking:

```
sasha or gloria and sunset <= snack or ken
```

Select an equivalent expression below so that Barbie's decision logic becomes clear and matches how Python would evaluate it:

Select one alternative:

- ☐ sasha or (gloria and ((sunset <= snack) or ken))
- ☐ (sasha or (gloria and (sunset <= snack))) or ken
- ☐ (((sasha or gloria) and sunset) <= snack) or ken
- ☐ sasha or (gloria and (sunset <= (snack or ken)))
- ☐ ((sasha or gloria) and (sunset <= snack)) or ken
- ☐ ((sasha) or (gloria)) and sunset <= (snack or ken)

Maximum marks: 2

1(o)

```
bags = 42
brand = "Norwegian Rain"
price_drop = -15.75
display = [20, "Tote bag", 3]
```

```
a = bags + bags
b = display + [1]
c = bags * brand
d = bool(bags - price_drop)
```

Lisa is a design intern at a recognized **fashion boutique** that specializes in outerwear. She's helping the team by using Python to track items from their spring capsule drop - a mix of statement pieces and bold accessories:

Assume that the code snippet above has been executed. What would be the datatypes of these variables? If the program would crash on the line, just write Error.

Select datatype for the variable

	int	float	str	bool	list	- ERROR -
a	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
b	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
c	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
d	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maximum marks: 2



1(p)

```
def motion_range_improve(R0):
    R = R0 + 10
    return R

first_throw = 20
second_throw = motion_range_improve(first_throw)
print(first_throw + second_throw)
```

Kristian is practicing in **projectile motion** modeling. For this purpose, he went to throw a rugby ball on the street. After collecting some data, he went back home and wrote the following programme:

What does this programme print? (if the programme crashes, write only Error)

Answer:

Maximum marks: 2

1(q)

```
def spa_schedule_switch(spa_services):

    for service in spa_services:
        new_status = 0
        if spa_services[service] == 0:
            new_status = 1
        spa_services[service] = new_status

spa_masters = {
    'Manicurist': 2,
    'Waxing Specialist': 0,
    'Eyebrow Technician': 1,
    'Skincare Specialist': 1,
    'Massage Therapist': 0,
}

weekend_services = spa_schedule_switch(spa_masters)
```

Nora is managing the booking system at her **Spa Treatment** Centre. During weekdays, certain specialists are available and others are not. On weekends, the spa offers a different set of services to give staff some rest, and clients - a broader range of treatments.

Here is the code she uses:

Select the value that would be printed after running these statements:

	None	ERROR	0	1	2
print(weekend_services)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
print(spa_masters['Manicurist'])	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
print(spa_masters['Massage Therapist'])	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
print(spa_masters['Manicurist']['Skincare Specialist'])	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
print(spa_masters['Skincare Specialist'])	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
print(weekend_services['Manicurist'])	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maximum marks: 3

1(r)

```

planks = 96
logo = "WoodyWorld"
clients = {102, 104, 107}
discount = -4.5
table = [120, "Skrivebord", 4]

```

A digital system is used to track **workshop inventory**, client orders, and production specs for a big construction shop. The previous contractor has messed up the code, mixing many expressions together. Henrik has been asked to sort out this mess. Help him to *determine the data type of each expression*.

Select datatype for each of the expression

	int	list	float	-- ERROR --	str	bool
planks // planks	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
len(clients)	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
clients[2]	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
"bord" in table	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
table[1][table[2]]	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
[clients]	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Maximum marks: 3

2(a)

```

from uib_inf100_graphics.event_app import run_app

def app_started(app):
    app.rows = 20
    app.cols = 9
    app.margin = 20
    # acne locations (col, row)
    app.acne_data = [(2, 2), (3, 3), (6, 4), (5, 16), (7, 18)]

def get_cell_bounds(app, row, col):
    grid_width = app.width - 2 * app.margin
    grid_height = app.height - 2 * app.margin
    cell_width = grid_width / app.cols
    cell_height = grid_height / app.rows
    x0 = app.margin + col * cell_width
    y0 = app.margin + row * cell_height
    x1 = x0 + cell_width
    y1 = y0 + cell_height
    return (x0, y0, x1, y1)

def is_invalid_location(app, row, col):
    is_along_side = col == 0 or col == app.cols - 1
    is_near_top = row < 4
    is_near_bottom = row > app.rows - 5
    is_in_left_eye = col in [2, 3] and row in [7, 8]
    is_in_right_eye = col in [5, 6] and row in [7, 8]
    return is_along_side and (is_near_top or is_near_bottom) \
        or is_in_left_eye \
        or is_in_right_eye

def redraw_all(app, canvas):
    for row in range(app.rows):
        for col in range(app.cols):
            if is_invalid_location(app, row, col):
                continue

            (x0, y0, x1, y1) = get_cell_bounds(app, row, col)
            color = "red" if (col, row) in app.acne_data else "#ffc27d"
            canvas.create_oval(x0, y0, x1, y1, fill=color, outline="black")

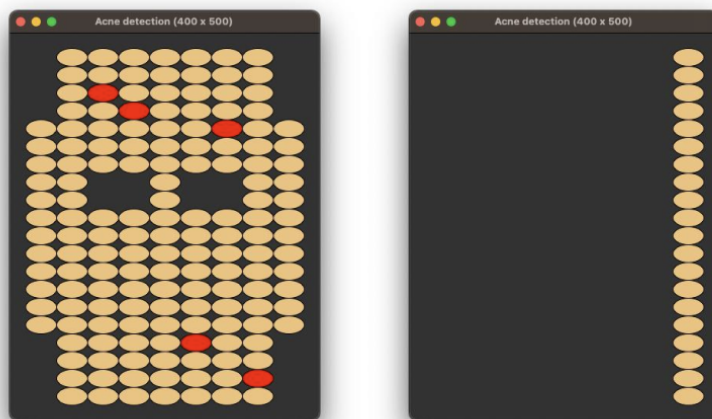
run_app(width=400, height=500, title="Acne detection")

```

Sofie is working on a **skincare** startup app that uses a simple face scanner to visualize **acne spots**. The visualization is built using a grid of "pores" zones (circles), where acne is marked in red. The app uses an `uib_inf100_graphics` graphics library.

Unfortunately, she made a bug that results in her visualisation only drawing the last column of the face. You can see the desired output of her code on the left, and the actual one on the right. The code is given under the pictures.

Your task is to help Sofie to solve the bug: write a detailed answer where the problem occurred, why did it happen, and how she can prevent this in further development.



Fill in your answer here

Format | B I U x<sub>2</sub> x<sup>2</sup> | I<sub>x</sub> | ↺ ↻ ↶ ↷ ↸ ↹ | ↵ ↶ ↷ ↸ ↹ | Ω ☐ | ✎ | Σ | ✖

Words: 0

Maximum marks: 9

**2(b)**

```
while stamina < exhaustion:
    still_fighting = TRUE
    print("The crowd goes wild! ")
```

Jax is building a **wrestling** league simulator in Python. Every fight has an 'exhaustion level' — once a wrestler hits that number, they lose the match. To simulate this, Jax uses a loop that keeps checking the wrestler's stamina and adds fatigue over time. You can see a simplified example of the loop above.

Jax expected this loop to eventually stop, but it doesn't. Instead, the match keeps going forever and eventually crashes Python. *What could be going wrong in the simulation? What should Jax keep in mind when designing this kind of logic?*

**Write your answer below.**

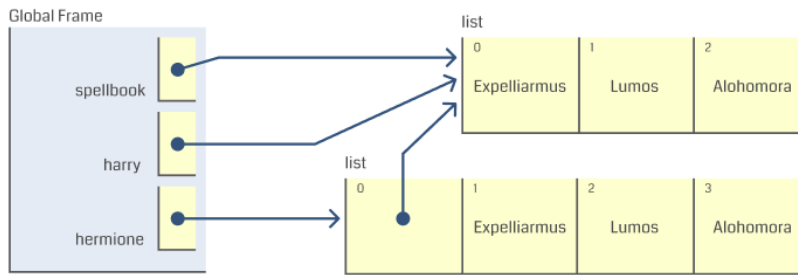
(Try to be both clear and specific in your explanation, as if you were helping a fellow student debug their code).

Format | **B** *I* U  $\times_2$   $\times^2$  |  $\int_x$  | | |

Words: 0

Maximum marks: 7

**3(a)** **Hermione and Harry Potter** are learning new spells from an old spell book: **Expelliarmus**, **Lumos**, and **Alohomora**. Part of their learning process is to write down learned spells in their own books. The structure of books and spells is shown in the picture below:



Write a code snippet that will correctly defines these variables and the memory state. You will not be deducted points if you also define other variables.

**Fill in your answer here**

Format

**B**


*I*


U


$\times_g$


$\times^2$


$I_x$


















































Words: 0

Maximum marks: 6

14/15