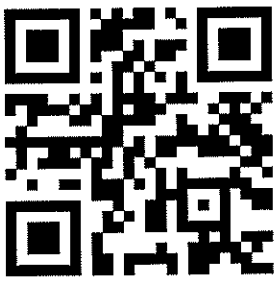

(d) **[3 marks] Memory model.**

Suppose we define the following function in a Python file, and then run the file in the Python console.

```
def my_function(n: int, numbers: list[int]) -> int:
    """Description omitted.

    Preconditions:
        - numbers != []
    """
    first_num = numbers[0]

    # MARKER A
    return n * first_num
```

(i) Now suppose we execute the following code in the Python console:

```
>>> x = 10
>>> my_nums = [2, x // 2]
>>> my_function(x + 1, my_nums)
```

Complete the *value-based memory model diagram* below to show the state of the variables when MARKER A is reached, immediately before the return statement in `my_function`. Not all rows may be used.

`__main__` **(Python console)**

Variable	Value
x	10
my_nums	[2,5]

`my_function`

Variable	Value
n	11
numbers	[2, 5]
first_num	2

(ii) Suppose we have executed the code given in Part (i), and then attempt to evaluate variable `first_num` in the Python console.

```
>>> my_function(x + 1, my_nums) # continuation of console display from Part (i)
22
>>> first_num
```

If this expression is evaluated successfully, write the value that is displayed in the Python console. If there is an error when this expression is evaluated, explain why the error occurs.

There is an error because the variable first_num is local and defined only inside my_function. Since we are calling it in __main__, we get an error because it does not exist here.