Documentation

Project 1 – weeks 4-6 – Stefan Lucian Gramada

Task: Write a program that can work with complex numbers (a + bi). The program will manage a list of complex numbers and allows to repeatedly perform actions like:

1. Add number to the list
   * Add number at the end of the list
   * Add number at a given position
2. Modify elements from the list
   * Eliminate number from a specific position
   * Eliminate numbers from an interval of positions
   * Replace all appearances of a number with another number
3. Search numbers
   * Print the imaginary part of the numbers from a subsequence
   * Print all the numbers which abs(x) < 10
   * Print all the numbers which abs(x) == 10
4. Operate with numbers
   * Sum of the numbers from a subsequence
   * Product of the numbers from a subsequence
   * Print the list descendent by the imaginary part
5. Filter
   * Eliminate numbers where the real part is prime
   * Eliminate numbers where the absolute value of them is <, = or > than a given number
6. Undo

Iteration Table:

|  |  |
| --- | --- |
| Iteration | Functions |
| I1 | 1.Add numbers to the list  3.Search numbers |
| I2 | 2.Modify elements from the list  5.Filter |
| I3 | 4.Operate with numbers  6.Undo  Implement final interface |

# Iteration 1

## Runtime scenarios

### Runtime Scenario for Add Action

|  |  |  |  |
| --- | --- | --- | --- |
|  | User | Program | Description |
| 1. |  | “Select your next action:  1.Add number  3.Search numbers  7.Exit” | Prints main UI |
| 2. | 1 |  | Select action 1 |
| 3. |  | “1.Add number in the list  2.Add number at a given position” | Prints action 1 UI |
| 4. | 1 |  | Select sub-action 1 |
| 5. |  | “Please enter your number x = a + bi, where:  The real part is a:” | Waits value for a |
| 6. | 2 |  | a becomes 2 |
| 7. |  | “The imaginary part is b:” | Waits value for b |
| 8. | -0.151 |  | b becomes -0.151 |
| 9. |  | “[(2.0, -0.151)]” | The complex number 2 - 0.151\*i is added to list |
| 10. |  | “Select your next action:  1.Add number  3.Search numbers  7.Exit” | Prints main UI |
| 11. | 1 |  | Select action 1 |
| 12. |  | “1.Add number in the list  2.Add number at a given position” | Prints action 1 UI |
| 13. | 2 |  | Select sub-action 2 |
| 14. |  | “Please enter your number x = a + bi, where:  The real part is a:” | Waits value for a |
| 15. | 0.3154 |  | a becomes 0.3154 |
| 16. |  | “The imaginary part is b:” | Waits value for b |
| 17. | 9 |  | b becomes 9 |
| 18. |  | “Enter position: “ | Waits for position |
| 19. | 0 |  | Position becomes 0 |
| 20. |  | “[(0.3154, 9.0), (2.0, -0.151)]” | The complex number 0.3154 + 9\*i is added at position 0 |

### Runtime Scenarios for Search Action

Let’s say that your list = [(0, 0), (7.071, -7.071), (-3.13, 0.921)]

|  |  |  |  |
| --- | --- | --- | --- |
|  | User | Program | Description |
| 1. |  | “Select your next action:  1.Add number  3.Search numbers  7.Exit” | Prints main UI |
| 2. | 3 |  | Select action 3 |
| 3. |  | “1.Print the imaginary part of the numbers from a subsequence  2.Print all the numbers which abs(x) < 10  3.Print all the numbers which abs(x) == 10” | Prints action 3 UI |
| 4. | 1 |  | Select sub-action 1 |
| 5. |  | “Enter the start and the end of your subsequence:” | Waits for the start and the end of the subsequence |
| 6. | 1 2 |  | start becomes 1 and end becomes 2 |
| 7. |  | “[-7.071, 0.921]” | Prints the imaginary part of the numbers from the subsequence start-end |
| 8. |  | “Select your next action:  1.Add number  3.Search numbers  7.Exit” | Prints main UI |
| 9. | 3 |  | Select action 3 |
| 10. |  | “1.Print the imaginary part of the numbers from a subsequence  2.Print all the numbers which abs(x) < 10  3.Print all the numbers which abs(x) == 10” | Prints action 3 UI |
| 11. | 2 |  | Select sub-action 2 |
| 12. |  | “(0.0, 0.0) (-3.13, 0.921)” | Prints the numbers from the list where abs(x) < 10 |
| 13. |  | “Select your next action:  1.Add number  3.Search numbers  7.Exit” | Prints action 3 UI |
| 14. | 3 |  | Select action 3 |
| 15. |  | “1.Print the imaginary part of the numbers from a subsequence  2.Print all the numbers which abs(x) < 10  3.Print all the numbers which abs(x) == 10” | Prints action 3 UI |
| 16. | 3 |  | Select sub-action 3 |
| 17. |  | (7.071, -7.071) | Prints the numbers from the list where abs(x) == 10 |

## Test Scenarios

### Test Scenarios for abs\_value(complex\_numbers)

|  |  |
| --- | --- |
| Data: complex number – tuple (a, b) | Results: float number with 3 zecimals |
| (1, 3) | 3.162 |
| (7.071, -7.071) | 9.999 |

### Test Scenarios for add\_number\_to\_list(complex\_number, list)

|  |  |
| --- | --- |
| Data: complex number – tuple(a, b), list | Results: adds complex number at the end of the list |
| (3.14, 0.159), [] | [(3.14, 0.159)] |
| (1, 0), [(3.14, 0.159)] | [(3.14, 0.159), (1, 0)] |

### Test Scenarios for add\_number\_to\_list\_position(complex\_number, list, position)

|  |  |
| --- | --- |
| Data: complex number – tuple(a, b), list, position | Results: adds complex number at a given position |
| (0, 0), [], 0 | [(0, 0)] |
| (1, 2), [(0,1), (23,87), (34, 87), (23, 567)], 1 | [(0,1), (1,2), (23,87), (34, 87), (23, 567)] |

### Test Scenarios for create\_complex\_number(real\_part, imaginary\_part)

|  |  |
| --- | --- |
| Data: real\_part – float,  imaginary\_part - float | Results: returns complex number as a tuple |
| 3.14, 0.159 | (3.14, 0.159) |

### Test Scenarios for create\_imaginary\_list(list, start, end, im\_list)

|  |  |
| --- | --- |
| Data: list, start, end, im\_list – list with the imaginary part of complex numbers | Results: adds in im\_list the imaginary part of the numbers from the subsequence start-end |
| [(0, 0), (1, 43), (-4, 0.135), (-4,96)]  1, 3, [] | [43, 0.135, 96] |

### Test Scenarios for validate\_position(position, list)

|  |  |
| --- | --- |
| Data: position, list | Results: Nothing or Error |
| 1, [(0,0), (1,1), (2,2)] | - |
| 3, [(0,0), (1,1), (2,2)] | IndexError("Position doesn't exist!") |

### Test Scenarios for validate\_subsequence(list, start, end)

|  |  |
| --- | --- |
| Data: list, start, end | Results: Nothing or Error |
| [(0,0) , (1,1), (2,2), (3, 3)], 1, 3 | - |
| [(0,0) , (1,1), (2,2), (3, 3)], -1, 10 | IndexError("Invalid subsequence!") |