Dallin Olson

Verdon Walker

CSE 131

November 5th, 2022

Lab 08: Sub-list Sort Analysis

MODULARIZATION METRICS

I would define the relationship between *select_mode* and *main* as being strong and encapsulated. Though I have not written the code yet, the only parameter that is returned to main is the final array which would have been vetted for errors prior to returning the array data and is only designed to select the form of data entry.

The metric of *manual_entry* would also be extraneous and simple given the potential for human error when it comes to imputing data. Data may be valid, but could not always be convenient or easy to understand. Also, the function is not necessary for the completion of the main purpose of the program. It only exists to meet the descriptive requirement for " a program which will prompt the user for the list of names or numbers."

The *test_cases* and its callee functions *get_filename* and *read_file* are strong and encapsulated. The purpose of the *test_cases* function is to automatically run the test cases without the need of user input. Each filename will be built in as a variable, read, and run with several checks to validate the data. I believe it is efficient and easy to understand.

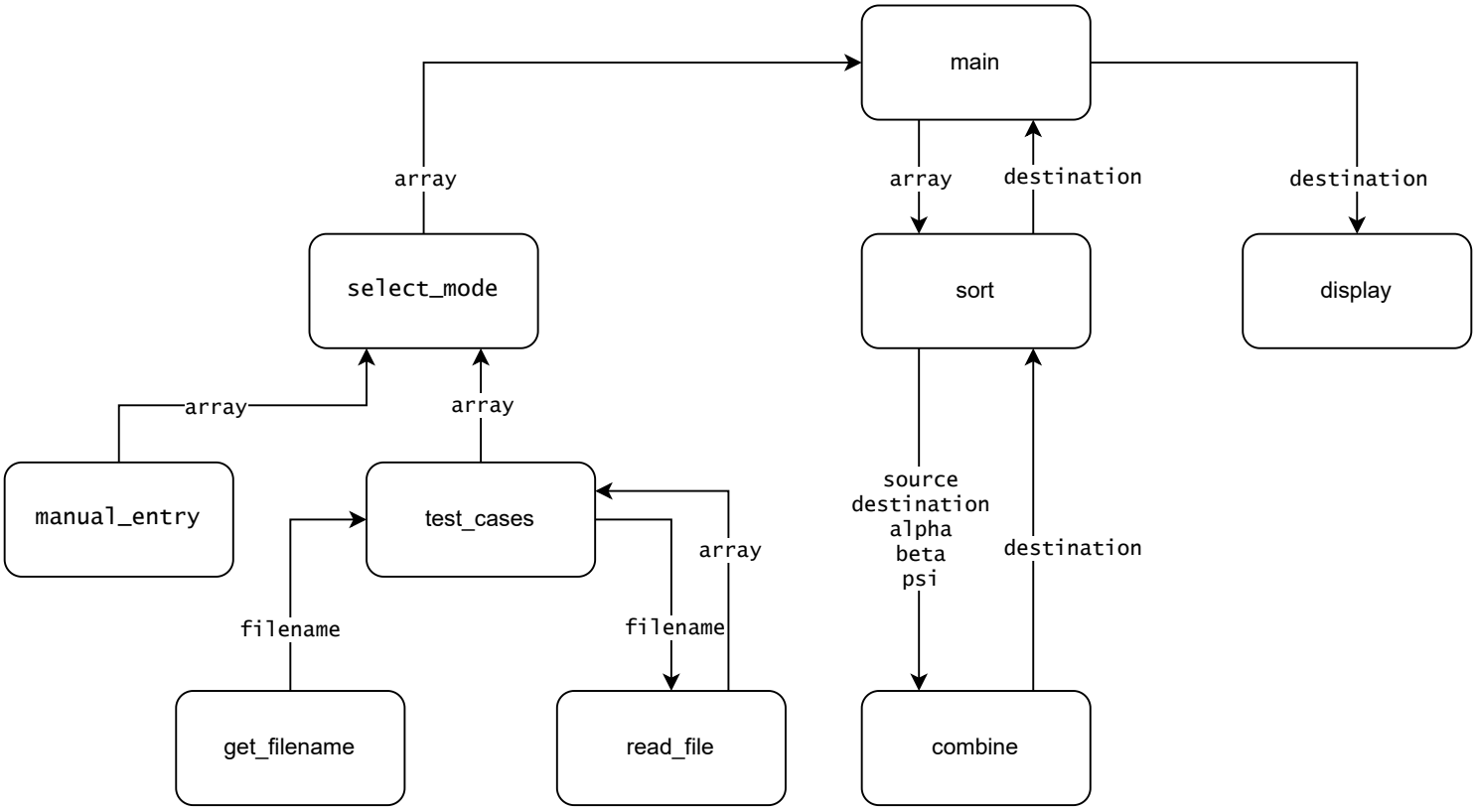
The main functions of the program, *sort* and *combine* are also strong and encapsulated with is backed by its O(n*logn) efficiency. Each parameter being communicated between the functions is necessary and always valid.

Finally, *display* is also strong and encapsulated. It has one purpose, which is to display the result and only requires the array parameter.

Overall I believe my structure charts show a strong and encapsulated design. However the coupling level is dependent upon the ability to validate data as it is passed between functions. Therefore it has the potential to slip to simple and perhaps extraneous.

ALGORITHM METRICS

I struggle with analyzing the O notation on a line by line basis. The majority of the lines are O(1). For the *sort* function, each *while* loop is O(log*n). While not exactly a log*n execution, each *while* loop follows that pattern. Again, this is based on a line by line analysis. The overall efficiency of the *sort* function is O(log*n). The *combine* function contains an O(n) because we iterate through our range in order to merge the lists resulting in an overall efficiency of O(n). The entire algorithm results in a O(n*(log*n)) by combining the notations of each function.



*Test cases will be
stored as json files
and run automatically

Name	P/C	Input	Output	Rationale
Non-List Parameter	N/A	0	Error	Input must be type list.
Empty List	N/A	[]	[]	No items to sort.
Integer List	N/A	[5, 4, 3, 2, 1]	[1, 2, 3, 4, 5]	Valid array.
Float List	N/A	[1.5, ... 1.1]	[1.1, ... 1.5	Valid array.
string List	N/A	['C', 'B', 'A']	['A', 'B', 'C']	Valid array.
Mixed List	N/A	[10, 'A', 5.5]	Error	Cant compare int to str.
Large List	N/A	[100, ... , 1]	[1, ... , 100]	Valid array.

```
      def sort(array)
01          size <- len(array)
02          source <- array
03          destination <- [None] * size
04          iteration <- 2

05          WHILE iteration > 1
06                iteration <- 0
07                begin1 <- 0
08                WHILE begin1 < size
09                      end1 <- begin1 + 1
10                      WHILE end1 < size and source[end1 - 1] <= source[end1]
11                            end1 +=1

11                      begin2 <- end1

12                      IF begin2 < size
13                            end2 <- begin2 + 1
14                      ELSE
15                            end2 <- begin2
16                      WHILE end2 < size and source[end2 - 1] <= source[end2]
17                      end2 += 1

18                      iteration += 1
19                      combine(source, destination, begin1, begin2, end2)
20                      begin1 <- end2
21                source, destination <- destination, source
22          RETURN source
      END

      def combine(source, destination, begin1, begin2, end2)
23          end1 <- begin2

24          FOR i in range(begin1, end2)
25                IF (begin1 < end1) and (begin2 == end2
                        or source[begin1] < source[begin2])

26                      destination[i] <- source[begin1]
27                      begin1 += 1
28                else
29                      destination[i] <- source[begin2]
30                      begin2 += 1
31          RETURN
      END
```

```
      def sort(array)
o(1)        size <- len(array)
o(1)        source <- array
o(1)        destination <- [None] * size
o(1)        iteration <- 2

o(log*n)    WHILE iteration > 1
o(1)              iteration <- 0
o(1)              begin1 <- 0
o(log*n)          WHILE begin1 < size
o(1)                    end1 <- begin1 + 1
o(log*n)                WHILE end1 < size and source[end1 - 1] <= source[end1]
o(1)                          end1 +=1

o(1)                    begin2 <- end1

o(1)                    IF begin2 < size
o(1)                          end2 <- begin2 + 1
o(1)                    ELSE
o(1)                          end2 <- begin2
o(log*n                 WHILE end2 < size and source[end2 - 1] <= source[end2]
o(1)                          end2 += 1

o(1)                    iteration += 1
o(1)                    combine(source, destination, begin1, begin2, end2)
o(1)                    begin1 <- end2
o(1)              source, destination <- destination, source
o(1)        RETURN source
      END

      def combine(source, destination, begin1, begin2, end2)
o(1)        end1 <- begin2

o(n)        FOR i in range(begin1, end2)
o(1)              IF (begin1 < end1) and (begin2 == end2
                        or source[begin1] < source[begin2])

o(1)                    destination[i] <- source[begin1]
o(1)                    begin1 += 1
o(1)              ELSE
o(1)                    destination[i] <- source[begin2]
o(1)                    begin2 += 1
o(1)        RETURN
      END
```

LINE	ARRAY	SIZE	source	DESTINTATION	ITERATION	BEGIN1	END1	BEGIN2	END2	I
01	[5, 4, 3, 2, 1]									
02	[5, 4, 3, 2, 1]									
03	[5, 4, 3, 2, 1]	5								
04	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]							
05	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]						
07	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	2					
08	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	2					
09	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0					
10	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0				
11	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0				
12	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1			
15	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1			
17	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1	1		
18	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1	1		
21	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1	1	2	
24	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	0	0	1	1	2	
25	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[N, N, N, N, N]	1	0	1	1	2	
31			[5, 4, 3, 2, 1]	[N, N, N, N, N]		0	1	1	2	
32			[5, 4, 3, 2, 1]	[N, N, N, N, N]		0	1	1	2	
34			[5, 4, 3, 2, 1]	[N, N, N, N, N]		0	1	1	2	
35			[5, 4, 3, 2, 1]	[N, N, N, N, N]		0	1	1	2	0
39			[5, 4, 3, 2, 1]	[N, N, N, N, N]		0	1	1	2	0
40			[5, 4, 3, 2, 1]	[4, N, N, N, N]		0	1	1	2	0
34			[5, 4, 3, 2, 1]	[4, N, N, N, N]		0	1	2	2	0
35			[5, 4, 3, 2, 1]	[4, N, N, N, N]		0	1	2	2	1
36			[5, 4, 3, 2, 1]	[4, N, N, N, N]		0	1	2	2	1
37			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		0	1	2	2	1
34			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		1	1	2	2	1
41			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		1	1	2	2	1
41			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		1	1	2	2	1
26	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	0	1	1	2	
10	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	1	1	2	
11	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	1	1	2	
12	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	1	2	
15	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	1	2	
17	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	3	2	
18	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	3	2	
21	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	3	4	
24	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	1	2	3	3	4	
25	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, N, N, N]	2	2	3	3	4	
31			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		2	3	3	4	
32			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		2	3	3	4	
34			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		2	3	3	4	
35			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		2	3	3	4	2
39			[5, 4, 3, 2, 1]	[4, 5, N, N, N]		2	3	3	4	2
40			[5, 4, 3, 2, 1]	[4, 5, 2, N, N]		2	3	3	4	2
34			[5, 4, 3, 2, 1]	[4, 5, 2, N, N]		2	3	4	4	2
35			[5, 4, 3, 2, 1]	[4, 5, 2, N, N]		2	3	4	4	3
36			[5, 4, 3, 2, 1]	[4, 5, 2, N, N]		2	3	4	4	3
37			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		2	3	4	4	3
34			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		3	3	4	4	3
41			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		3	3	4	4	3
41			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		3	3	4	4	3
26	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	2	3	3	4	
10	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	3	3	4	
11	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	3	3	4	
12	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	3	4	
15	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	3	4	
17	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	5	4	
20	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	5	4	
21	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	5	5	
24	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	2	4	5	5	5	
25	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]	3	4	5	5	5	
31			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		4	5	5	5	
32			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		4	5	5	5	
34			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		4	5	5	5	
35			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		4	5	5	5	4
36			[5, 4, 3, 2, 1]	[4, 5, 2, 3, N]		4	5	5	5	4
37			[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]		4	5	5	5	4

37		[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]		4	5	5	5	4	
34		[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]		5	5	5	5	4	
41		[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]		5	5	5	5	4	
41		[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]		5	5	5	5	4	
26	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]	3	4	5	5	5	
10	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]	3	5	5	5	5	
27	[5, 4, 3, 2, 1]	5	[5, 4, 3, 2, 1]	[4, 5, 2, 3, 1]	3	5	5	5	5	
07	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	3	5	5	5	5	
08	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	3	5	5	5	5	
09	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	5	5	5	5	
10	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	5	5	5	
11	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	5	5	5	
12	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	1	5	5	
13	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	1	5	5	
12	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	5	5	
15	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	5	5	
17	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	5	
18	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	5	
21	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	3	
22	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	3	
21	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	4	
24	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	0	0	2	2	4	
25	[5, 4, 3, 2, 1]	5	[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]	1	0	2	2	4	
31			[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]		0	2	2	4	
32			[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]		0	2	2	4	
34			[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]		0	2	2	4	
35			[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]		0	2	2	4	0
39			[4, 5, 2, 3, 1]	[5, 4, 3, 2, 1]		0	2	2	4	0
40			[4, 5, 2, 3, 1]	[2, 4, 3, 2, 1]		0	2	2	4	0
34			[4, 5, 2, 3, 1]	[2, 4, 3, 2, 1]		0	2	3	4	0
35			[4, 5, 2, 3, 1]	[2, 4, 3, 2, 1]		0	2	3	4	1
39			[4, 5, 2, 3, 1]	[2, 4, 3, 2, 1]		0	2	3	4	1
40			[4, 5, 2, 3, 1]	[2, 3, 3, 2, 1]		0	2	3	4	1
34			[4, 5, 2, 3, 1]	[2, 3, 3, 2, 1]		0	2	4	4	1
35			[4, 5, 2, 3, 1]	[2, 3, 3, 2, 1]		0	2	4	4	2
36			[4, 5, 2, 3, 1]	[2, 3, 3, 2, 1]		0	2	4	4	2
37			[4, 5, 2, 3, 1]	[2, 3, 4, 2, 1]		0	2	4	4	2
34			[4, 5, 2, 3, 1]	[2, 3, 4, 2, 1]		1	2	4	4	2
35			[4, 5, 2, 3, 1]	[2, 3, 4, 2, 1]		1	2	4	4	3
36			[4, 5, 2, 3, 1]	[2, 3, 4, 2, 1]		1	2	4	4	3
37			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		1	2	4	4	3
34			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		2	2	4	4	3
41			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		2	2	4	4	3
41			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		2	2	4	4	3
26	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	0	2	2	4	
10	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	2	2	4	
11	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	2	2	4	
12	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	2	4	
15	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	2	4	
17	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	5	4	
20	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	5	4	
21	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	5	5	
24	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	1	4	5	5	5	
25	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	2	4	5	5	5	
31			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	
32			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	
34			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	
35			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
36			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
37			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
34			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
41			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
41			[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]		4	5	5	5	4
26	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	2	4	5	5	5	
10	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	2	5	5	5	5	
27	[2, 3, 4, 5, 1]	5	[4, 5, 2, 3, 1]	[2, 3, 4, 5, 1]	2	5	5	5	5	
07	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	2	5	5	5	5	

07	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	2	5	5	5	5		
08	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	2	5	5	5	5		
09	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	5	5	5	5		
10	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	5	5	5		
11	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	5	5	5		
12	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	1	5	5		
13	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	1	5	5		
12	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	2	5	5		
13	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	2	5	5		
12	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	3	5	5		
13	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	3	5	5		
12	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	5	5		
15	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	5	5		
17	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	4	5		
18	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	4	5		
21	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	4	5		
24	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	0	0	4	4	5		
25	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]	1	0	4	4	5		
31			[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]		0	4	4	5		
32			[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]		0	4	4	5		
34			[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]		0	4	4	5		
35			[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]		0	4	4	5		0
39			[2, 3, 4, 5, 1]	[4, 5, 2, 3, 1]		0	4	4	5		0
40			[2, 3, 4, 5, 1]	[1, 5, 2, 3, 1]		0	4	4	5		0
34			[2, 3, 4, 5, 1]	[1, 5, 2, 3, 1]		0	4	5	5		0
35			[2, 3, 4, 5, 1]	[1, 5, 2, 3, 1]		0	4	5	5		1
36			[2, 3, 4, 5, 1]	[1, 5, 2, 3, 1]		0	4	5	5		1
37			[2, 3, 4, 5, 1]	[1, 2, 2, 3, 1]		0	4	5	5		1
34			[2, 3, 4, 5, 1]	[1, 2, 2, 3, 1]		1	4	5	5		1
35			[2, 3, 4, 5, 1]	[1, 2, 2, 3, 1]		1	4	5	5		2
36			[2, 3, 4, 5, 1]	[1, 2, 2, 3, 1]		1	4	5	5		2
37			[2, 3, 4, 5, 1]	[1, 2, 3, 3, 1]		1	4	5	5		2
34			[2, 3, 4, 5, 1]	[1, 2, 3, 3, 1]		2	4	5	5		2
35			[2, 3, 4, 5, 1]	[1, 2, 3, 3, 1]		2	4	5	5		3
36			[2, 3, 4, 5, 1]	[1, 2, 3, 3, 1]		2	4	5	5		3
37			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 1]		2	4	5	5		3
34			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 1]		3	4	5	5		3
35			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 1]		3	4	5	5		4
36			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 1]		3	4	5	5		4
37			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]		3	4	5	5		4
34			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]		4	4	5	5		4
41			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]		4	4	5	5		4
41			[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]		4	4	5	5		4
26	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]	1	0	4	4	5		
10	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]	1	5	4	4	5		
27	[2, 3, 4, 5, 1]	5	[2, 3, 4, 5, 1]	[1, 2, 3, 4, 5]	1	5	4	4	5		
07	[2, 3, 4, 5, 1]	5	[1, 2, 3, 4, 5]	[2, 3, 4, 5, 1]	1	5	4	4	5		
28	[2, 3, 4, 5, 1]	5	[1, 2, 3, 4, 5]	[2, 3, 4, 5, 1]	1	5	4	4	5		
28	[2, 3, 4, 5, 1]	5	[1, 2, 3, 4, 5]	[2, 3, 4, 5, 1]	1	5	4	4	5		