Week 7 Project—Functional Programming Versus Object Oriented Programming

**Scenario**

In this week’s lab you will look at functional programming in scheme and compare it to Object Oriented programming.

**Rubric**

Point distribution for this activity:

|  |  |  |
| --- | --- | --- |
| **Lab Activity** | | |
| **Document** | **Points possible** | **Points received** |
| Part A | 25 |  |
| Part B | 20 |  |
| Part C | 5 |  |
| **Total Points** | **50** |  |

Part A: Functional Programming in scheme

Use a scheme interpreter online (or download an interpreter):

<https://repl.it/languages/Scheme/>

If that doesn’t work try -

<https://replit.com/new/scheme>

Create a new replit (note, you may have to log in)

Graphical user interface, text, application

Description automatically generated

We will look at two different functions in scheme and compare them to an OOP language.

1. A famous algorithm for sorting a list is called quicksort. Type in the following code for quick sort into the scheme interpreter.

(define qsort

(lambda (l)

(let ((lesser '()))

(let ((greater '()))

(cond

((null? l) '())

(else (map (lambda (ele)

(if (> (car l) ele)

(set! lesser (cons ele lesser))

(set! greater (cons ele greater)))) (cdr l))

(append (qsort lesser) (cons (car l) (qsort greater))))

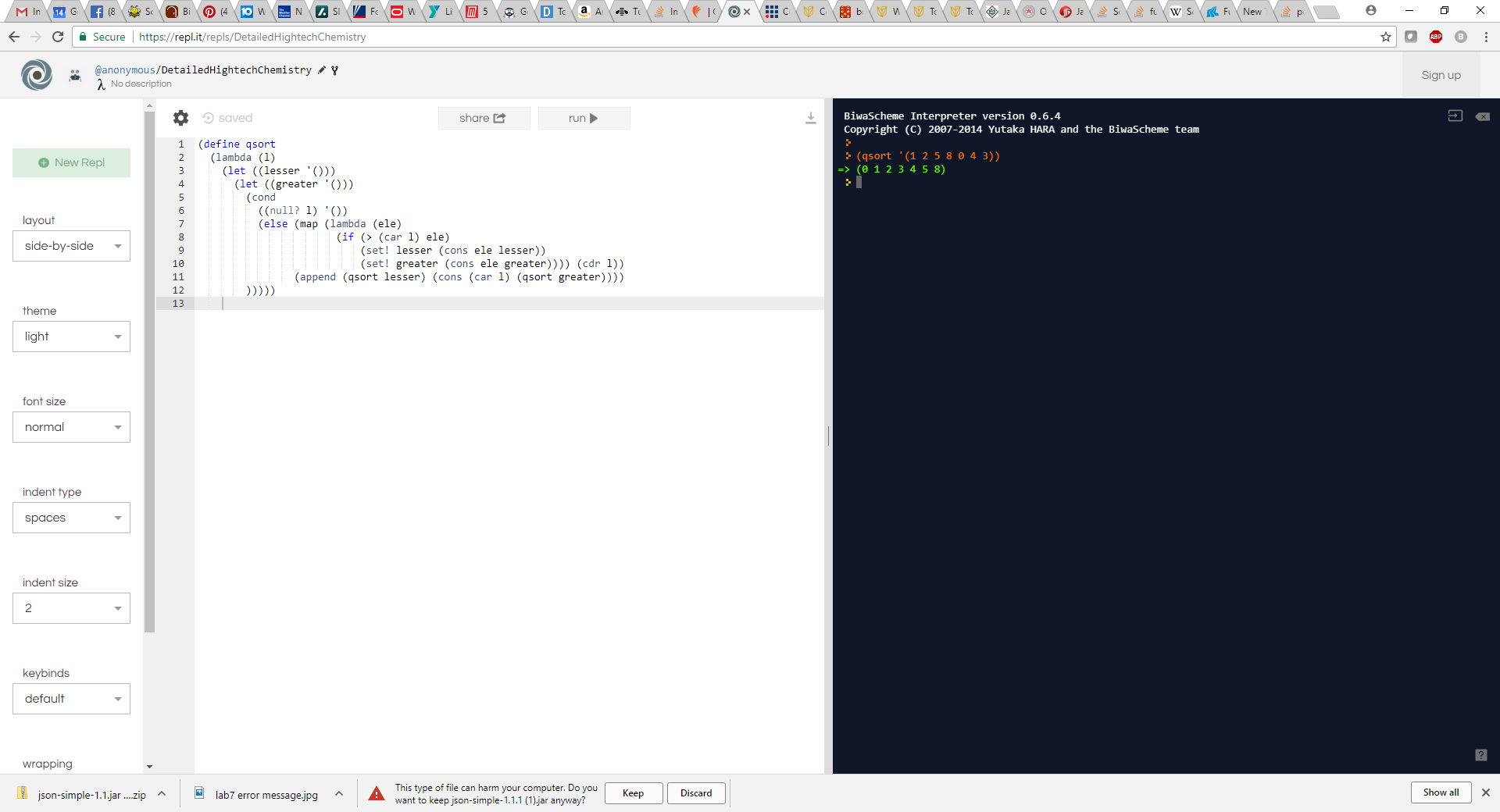
)))))

To run this code pick run at the top of the screen (note: the prompt will change to Stop)

Graphical user interface, application

Description automatically generated

Then type in the function name and a set of numbers as below to so that scheme will sort them. For example: (qsort ‘(1 2 5 8 0 4 3))



Add **different** numbers than above and paste a screenshot showing it working to sort the list. Then press Stop:

A screenshot of a computer

Description automatically generated

1. Another famous function that adds up the previous two numbers to create a list is called the Fibonacci sequence

Write this code in scheme to create a Fibonacci sequence (note the example screenshot shows the Fibonacci sequence for 11):

(define (fib n)

(if (< n 2) n (+ (fib (- n 1))

(fib (- n 2)))))

Graphical user interface, application, Teams

Description automatically generated

What is the result of the 10th element? And the 15th element?

(fib 10)

Answer: 55

(fib 15)

Answer: 610

Part B:

1. To examine the differences between the code, we will implement the same code in Java, C++ or C#.

Example Java Code:

Qsort class:

class Qsort {

/\* This function takes last element and places it in the correct position in the sorted array. Then places smaller elements to the left and greater to the right

\*/

int partition(int arr[], int low, int high)

{

int pivot = arr[high];

int i = (low-1); // index of smaller element

for (int j=low; j<high; j++)

{

// If current element is smaller than or

// equal to pivot element

if (arr[j] <= pivot)

{

i++;

// swap arr[i] and arr[j]

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

// swap array places

int temp = arr[i+1];

arr[i+1] = arr[high];

arr[high] = temp;

return i+1;

}

/\* The main function that implements QuickSort()

arr[] --> Array to be sorted,

low --> Starting index,

high --> Ending index \*/

void sort(int arr[], int low, int high)

{

if (low < high)

{

/\* pi is partitioning index, arr[pi] is

now at right place \*/

int pi = partition(arr, low, high);

// Recursively sort elements before

// partition and after partition

sort(arr, low, pi-1);

sort(arr, pi+1, high);

}

}

}

Quicksort class (including main):

public class Quicksort {

public static void main(String[] args) {

int arr[] = {1, 2, 5, 8, 0, 4, 3};

int n = arr.length;

Qsort ob = new Qsort();

ob.sort(arr, 0, n-1);

System.out.println("sorted array");

printArray(arr);

}

static void printArray(int arr[])

{

int n = arr.length;

for (int i=0; i<n; ++i)

System.out.print(arr[i]+" ");

System.out.println();

}

}

Test the array above (same as in scheme) and your own input:

Screenshot of output below:

A screenshot of a computer program

Description automatically generated

Screenshot of your own input for the array:

A screenshot of a computer

Description automatically generated

1. Implement Fibonacci sequence in Java, C# or C++

Java Example below:

public class FibSequence {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

int count = 16;

int[] feb = new int[count];

feb[0] = 0;

feb[1] = 1;

for(int i = 2; i < count; i++) {

feb[i] = feb[i-1] + feb[i-2];

}

//include code here to print out the 10th element (actually feb[10] and the 15th element)

}

}

Include screenshot of the code showing the results for printing the 10th element and the 15th elements:

A screenshot of a computer program

Description automatically generated

Part C:

What are 3 differences between functional programming and object oriented programming?

Functions vs Objects are the main focus

Recursion vs Loops for iteration

Support for parallel programming

Why would you choose to program in functional programming rather than OOP?

Functional programming excels when there is a fixed amount of items you are interacting with, and only add new operations as the program grows. As opposed to OOP, that support the growth of items as well as objects but can lead to more bugs.