# CS2613: Programming Languages Laboratory (FR02A)
## Lab #21 – Winter 2024

**Language:** Final Exam Review
**# of Tasks:** 4
**Topics:**

- Review for Final Exam

*All tasks are to be completed individually in line with the academic offense guidelines detailed on the syllabus and are **due before the end of the lab period** unless stated otherwise.*

**Task #1**

**Task Style:** Organization

**Description:**

During the final exam, you will not have access to the Internet beyond. It is recommended that you take some time to organize all your labs and programming questions and make sure you are not missing any files. Take some time to remember what each lab and programming question covered so you can quickly find resources as you need them. Feel free to make additional documents to help you organize your files.

**Task #2**

**Task Style:** Tracing

**Description:**

Trace each piece of code below and predict the output. When you think you have the right answers, run the code to check your answers.

Python:

```
mystery1 = (lambda p1, p2, p3: (p1[0] if (len(p3)%2 > 0) else (p2[len(p2)-1])))
print(mystery1("Hello", "World", "Test"))
print(mystery1("CS2613", "Final", "Exam!"))

mystery2 = (lambda v1: (v1 if (v1 < 0) else v1 * -1))
vals = map(mystery2, [4, -2, 14, -13, 3])
for val in vals:
	print(val)

mystery3 = (lambda m1: True if (m1 < 40) else False)
filtered = filter(mystery3, [89, 15, 135, 156, 156])
for val in filtered:
	print(val)
```

JavaScript:

```
const mystery1 = (message, fun) => {
	if(message.length > 6){
		let t = message.split(",");
		let s = 0;
		for(let i = 0; i < t.length; i++){
			s = s + parseInt(t[i])
		}
		return s;
	}
	else if(message.length >= 3){
		let m = message.substring(2) + 1 + "hello";
		return m;
	}
	else if(message.length >= 1){
		let m = fun(parseInt(message));
		return m;
	}

	return 0;
}

const mystery2 = (num) => {
	return num * num
}

console.log(mystery1("1,3,7,1,4,9", mystery2))
console.log(mystery1("608", mystery2))
console.log(mystery1("2", mystery2))
```

Racket:

```
(define (subtract x y)
  (cond [(<= (length y) 0) '()]
  [else (cons (- (list-ref x 0) (list-ref y 0)) (subtract (drop x 1) (drop y 1)))])
  )

(define (fun1 x) (/ x 3))
(define (fun2 x) (sqrt (+ x 1)))
(define (fun3 x) (>= x 0))

(define (fun4 a b c d e) (filter c (subtract (map a d) (map b e))))

(fun4 fun1 fun2 fun3 '(60 30 120 9) '(0 3 8 24))
```

Octave:

```
f1 = @(x, y) tolower(x)(y);
f2 = @(x, y) tolower(x)(length(x) + 1 - y);
f3 = @(x, y) x == y;

function retval = f4(m, f1, f2, f3)
	count = 0;
	range = 0;
	%mod is modulo division. Example: length(m) % 2 is equal to mod(length(m), 2)
	if(mod(length(m), 2) == 0)
		range = length(m)/2;
	else
		range = int32(length(m)/2)+1;
	end

	for i = 1:range
		if(f3(f1(m, i), f2(m, i)))
			count = count + 1;
		end
	end
	retval = count;

end


disp(f4("hello", f1, f2, f3));
disp(f4("bonjour!", f1, f2, f3));
disp(f4("781380485687", f1, f2, f3));
```

**Task #3**

**Task Style:** Programming

**Description:**

Write a Racket recursive function (direct or indirect) that takes no parameters and returns a list of all values [10, 20] after being calculated in the following piecewise function:

$$f(x) = \begin{cases} \frac{x}{2}, & if\ x\ is\ even \\ x * 2, & if\ x\ is\ odd \end{cases}$$

You may choose to do this with a purely functional programming approach or not. Review the functional programming Racket lab to review how to create a list of numbers in a range.

**Example:**

```
func_name() = [5, 22, 6, 26, 7, 30, 8, 34, 9, 38, 10]
```

## Task #4

**Task Style:** Coding

**Description:**

If there are topics you wish for more practice in a specific language, you may wish to attempt similar labs/programming questions that were designed for other languages.

For example, if you are nervous about file I/O in Python, you could try to create a Python solution for any of the labs for the other languages in Cycle 2, Octave 1, or Octave 3.