UMassAmherst

Manning College of Information
& Computer Sciences

Programming Methodology

Lab 13: Final Review

Wednesday, December 3rd, 2025

Weekly Lab Agenda

- Go over reminders/goals
- Review past material
- Work in groups of 2-3 to solve a few exercises
    - Lab leaders will assign new groups this week
- Discussion leaders will walk around and answer questions
- Solutions to exercises will be reviewed as a class
- Attendance taken at the end

Reminders

- Great job this semester everyone, you should be proud of your hard work!
  - Best of luck in these final couple of weeks!
- Final exam Logistics:
  - Tuesday, December 16th, 3:30pm - 5:30pm
- Please fill out the SRTI course survey, this really helps make the class better! You can find a link to this on Canvas.
- Homework 7 is due this Sunday (12/7) at 11:59pm
- Two more graded requirements for homework 6:
  - Homework 6 Team Evaluations are due today
  - Homework 6 Self Evaluations will be released by tomorrow and due later this week

Today’s Goals

- Review select topics for the final exam
  - Program correctness
  - Interpreters
  - Please review lectures, labs, and past exams for more practice on other topics covered this semester
- Fill out feedback form for 220 labs from this semester!
  - Please fill this out! We would love to hear from you as we aim to improve these labs in future semesters

What’s an Invariant?

- As its name implies, an invariant is something that *does not change*
    - Here: a *specific type of assertion* (not all assertions are invariants)
- We will work with **loop invariants**: they hold **before** each loop iteration (before checking the condition)
    - A useful invariant relates variables that *change* within the loop
    - The invariant will not hold at *every* point in the loop but will be *restored* (holds again) at the end

Exercise 1: Program Correctness

Write a function maxInInterval(A, start, stop), where **A** is an array of integers, **start** is an integer >= 0, and **stop** is an integer >= **start**. Both **start** and **stop** are assumed to be valid indices for array **A**.

maxInInterval(A, start, stop) returns the largest number found in array **A** between positions **start** and **stop**, both inclusive.

While writing this function, state your invariants. Show how your invariants change throughout your function. Try to use a while loop in your solution and show that the loop invariant is reestablished before each new iteration.

Note: *slightly* harder than what you can expect as an exam question!

Exercise 2: Interpreters

You are given an expression AST containing only numbers and the binary operators “+” and “-”. The given expression AST also contains exactly one variable node, corresponding to the variable x. The property containsX has also been added to each AST node, indicating if the variable x is within the AST whose root is that node.

This means we define the Expression type in this problem as follows:

```
type Expression = { kind: "number", value: number, containsX: false }
       | { kind: "variable", name: "x", containsX: true }
       | { kind: "operator", operator: “+” | “-”, left: Expression,
           right: Expression, containsX: boolean };
```

Exercise 2: Interpreters

Write a function solveForX(e: Expression, result: number): number that returns the value that x must be in order for the given expression **e** to evaluate to **result**.

You may use the function evalExpression(e: Expression), which will evaluate a given expression AST that does not contain any variables (though you should feel comfortable implementing this yourself too!).

```
type Expression = { kind: "number", value: number, containsX: false }
      | { kind: "variable", name: "x", containsX: true }
      | { kind: "operator", operator: “+” | “-”, left: Expression,
          right: Expression, containsX: boolean };
```

Anonymous Lab Feedback

https://forms.gle/iciZEVcuhF4UDNzQ6