UNIVERSITY AT ALBANY
State University of New York

COLLEGE OF ENGINEERING AND APPLIED SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

ICSI311 Principles of Programming Languages

Assignment 03 Created by Qi Wang

Table of Contents

Part I: General Information

- All assignments are individual assignments unless it is notified otherwise.
- All assignments must be submitted via Blackboard. No late submissions or e-mail submissions or hard copies will be accepted.
- Unlimited attempts will be allowed on Blackboard. **Only the last attempt will be graded.**
- Work will be rejected with no credit if
    - The work is late.
    - The work is not submitted properly (Blurry, wrong files, crashed files, etc.).
    - The work is a copy or partial copy of others' work (such as work from another person or the Internet).
- Students must turn in their original work. Any cheating violation will be reported to the college. Students can help others by sharing ideas and should not allow others to copy their work.
- Documents to be submitted:
    - Scheme source files
- Students are required to all the error-free source files with comments. Lack of any of the required items or programs with errors will result in a really low credit or no credit.
- **Grades and feedback**: TAs will grade. Feedback and grades for properly submitted work will be posted on Blackboard. Students have limited time/days from when a grade is posted to dispute the grade. Check email daily for the grade review notifications sent from the TAs. If students have any questions regarding the feedback or the grade, they should reach out to their TAs first.

Part II: Grading Rubric

ABET Student Outcomes:

SO2: Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.

Student Learning Outcomes:
SLO5: Be able to create small programs in Scheme

		Levels of Performance			
		UNSATISFACTORY	**DEVELOPING**	**SATISFACTORY**	**EXEMPLARY**
Performance Indicators	**Performance Indicator #1** The software meets the specification: • Use Scheme as a pure functional language • Use divide-and conquer and recursion properly • The program is tested completely • There are at least 10 test cases • Unallowed build-in functions are not used • more	The software doesn't meet the specification.	The software meets some of the specification.	The software meets all of the specification but there are minor issues.	The software meets all of the specification and handles all cases.
	Part I	0	/3	/5	/7
	Part II a	0	/3	/5	/7
	Part II b	0	/3	/5	/7
	Part III	0	/7	/10	/17
	Performance indicator #2 The code is well formatted and properly documented	The code is not well formatted and has minimal comments.	The program code is complete with minor issues and the code is somewhat formatted and has some comments.	The program code is complete with minor issues and the code is mostly formatted and properly documented.	The program code is complete with no issues and the code is properly formatted and documented.
	Part I	0	/1	/2	/3
	Part II a	0	/1	/2	/3
	Part II b	0	/1	/2	/3
	Part III	0	/1	/2	/3

Part III: Description

Assignment 3 Scheme Programming

Goals:

- Review and develop a deep and comprehensive understanding of functional programming
- Review and develop a deep and comprehensive understanding of the divide-and-conquer technique and programming technique such as recursion

Instructions:

In this assignment, you will complete four programs using Scheme. Please read the following requirements carefully:

- Use Scheme as a **pure** functional programming language.
- Use divide-and-conquer technique (If a function is big/not single-mined, write subfunctions.) and recursion programming technique.
- Each program must be tested completely. There must be at least 10 test cases. Include the test cases at the bottom of the program.
- Adequate comments must be provided for each logical block of a program. If a built-in function is used, the explanation of the function must be included. Assignments with minimal comments will be rejected with no credits. The following built-in functions are not allowed in this assignment:
  - reverse
  - max
  - min
  - map
  - apply
- Codes must be properly formatted.
- Save each part in its own program. Submit them all on Blackboard.

Part I (10 points):

abs is a built-in function that takes an argument and returns the absolute value of the argument. For example,

```
> (abs 3.22)
3.22
> (abs -5.16)
5.6
> (abs -2)
2
```

Write a function that takes a list as an argument and returns a list of absolute values of each of the elements. Assume the list will contain 0 or more numbers and no sublists. Built-in functions *map* or *apply* can't be used.

Part II (10 points each):

a) Write a function that takes an atom and a list as parameters and returns a list identical to its parameter list except with all instances of the given atom deleted. For example, the following shows four test cases.

An atom	A list	The retuned list
a	(a b c d a c a a)	(b c d c)
a	((a b) c () d a c a a)	((b) c () d c)

a	(((a (a)) b) c () d a c a a)	((((()) b) c () d c)
a	(a b c d a c (a ((a)))a a)	(b c d c ((())))

b) Write a function that takes two atoms and a list as parameters and returns a list identical to the parameter list except all occurrences of the first given atom in the list are replaced with the second given atom, no matter how deeply the first atom is nested.

Part III (20 points):

Write a bubble sort function that takes a list as the only parameter and returns a sorted list. Assume that there are no sublists in the original list.