CSc 600: PROGRAMMING LANGUAGE DESIGN **Dr. Jozo Dujmović**

Scheme

1. The concept of first class objects is fundamental for Scheme programming. In particular, in Scheme functions are first class objects. The main properties of functions as first class objects are exemplified by answering the following questions:
   (a) The first class object may be expressed as an anonymous literal value (constant). Show an example of the anonymous function and its use.
   (b) The first class object may be stored in variables (i.e. it may have a symbolic name). Show examples of defining and using named functions.
   (c) The first class object may be stored in data structures. Show an example of a data structure(e.g. a list) that contains functions
   (d) The first class object may be comparable to other objects for equality. Show an example of comparing functions and lists for equality.
   (e) The first class object may be passed as parameter to procedures/functions. Show an example of passing function as an argument to another function.
   (f) The first class object may be returned as result from procedures/functions. Show an example of returning a function as a result of another function.
   (g) The first class object may be readable and printable. Show examples of reading function(s) from the keyboard, reading function(s) from a file, and displaying a function.

2. [*This problem was used for the final exam*]

The mean value of n numbers is $\bar{x} = (x_1 + ... + x_n)/n$. The mean value of n squares is $\overline{x^2} = (x_1^2 + ... + x_n^2)/n$. The standard deviation is defined as $\sigma = \sqrt{\overline{x^2} - (\bar{x})^2}$. Write a Scheme function **sigma** that computes the standard deviation of any number of arguments, as in the following example:

```
[55] (sigma 1 2 3 2 1)
0.748331477354788
[56] (sigma 1 3 1 3 1 3)
1.
[57] (sigma 1 3)
1.
[58] (sigma 1)
0.
```

3. [*This problem was used for the final exam*]

(a) Write a recursive Scheme procedure **line** that prints n asterisks in a line as follows:

-
```
(line 5)
*****
```

(b) Write a recursive Scheme procedure **histogram** that uses the procedure **line**, and prints a histogram for a list of integers:

```
➢ (histogram '(1 2 3 3 2 1))
*
**
***
***
**
*
```

4. Write a Scheme program for computing a maximum of function f(x) within the interval [x1, x2]. Use the trisection method, and find the coordinate of maximum xmax with accuracy of 6 significant decimal digits.

5. Develop a program that computes the scalar product of two vectors. The program must not accept vectors having different size (in such a case print an error message). For example:

```
>(scalar-product '#(1 2 3) '#(2 1 1))
7
>(scalar-product '#(1 2 3) '#(1 2 3 4 5))
ERROR: Different sizes of vectors!
>
```

(a) Write the program in iterative style using the DO loop.

(b) Write the program using recursion.

6. The files "matrix1.dat" and "matrix2.dat" are created using a text editor and contain two rectangular matrices. For example,

matrix1.dat:

```
2 3
1 2 3
4 5 6
```

matrix2.dat:

```
3 3
1 2 3
1 2 3
1 2 3
```

In both cases the first row contains the size of the matrix (the number of rows and the number of columns). The remaining rows contain the values of elements.

(a) Develop programs **row** and **col** that read a matrix from a file and display a specified row

or column. For example:

```
> (row "matrix1.dat" 2)
4 5 6
> (col "matrix1.dat" 2)
2 5
```

Matrices should be stored in memory as vectors whose components are vectors.

(b) Develop a program for matrix multiplication **mmul** that multiplies two matrices stored in specified input files, and creates and displays an output file containing the product. For example:

```
> (mmul "matrix1.dat" "matrix2.dat" "matrix3.dat")
 6  12  18
15  30  45
```

In this example the contents of the new file "matrix3.dat" should be

```
 2   3
 6  12  18
15  30  45
```

NOTES: All Scheme programs must be presented as complete transcripts which include the headers with author name, program name, program description, date, and homework and problem numbers (e.g. HW1/4). Transcripts must include sample program outputs. All critical points in a program must be explained using comments. ***Late homework cannot be accepted.***