CS 460

Programming Languages

Fall 2023

Dr. Watts

(13 September 2023)

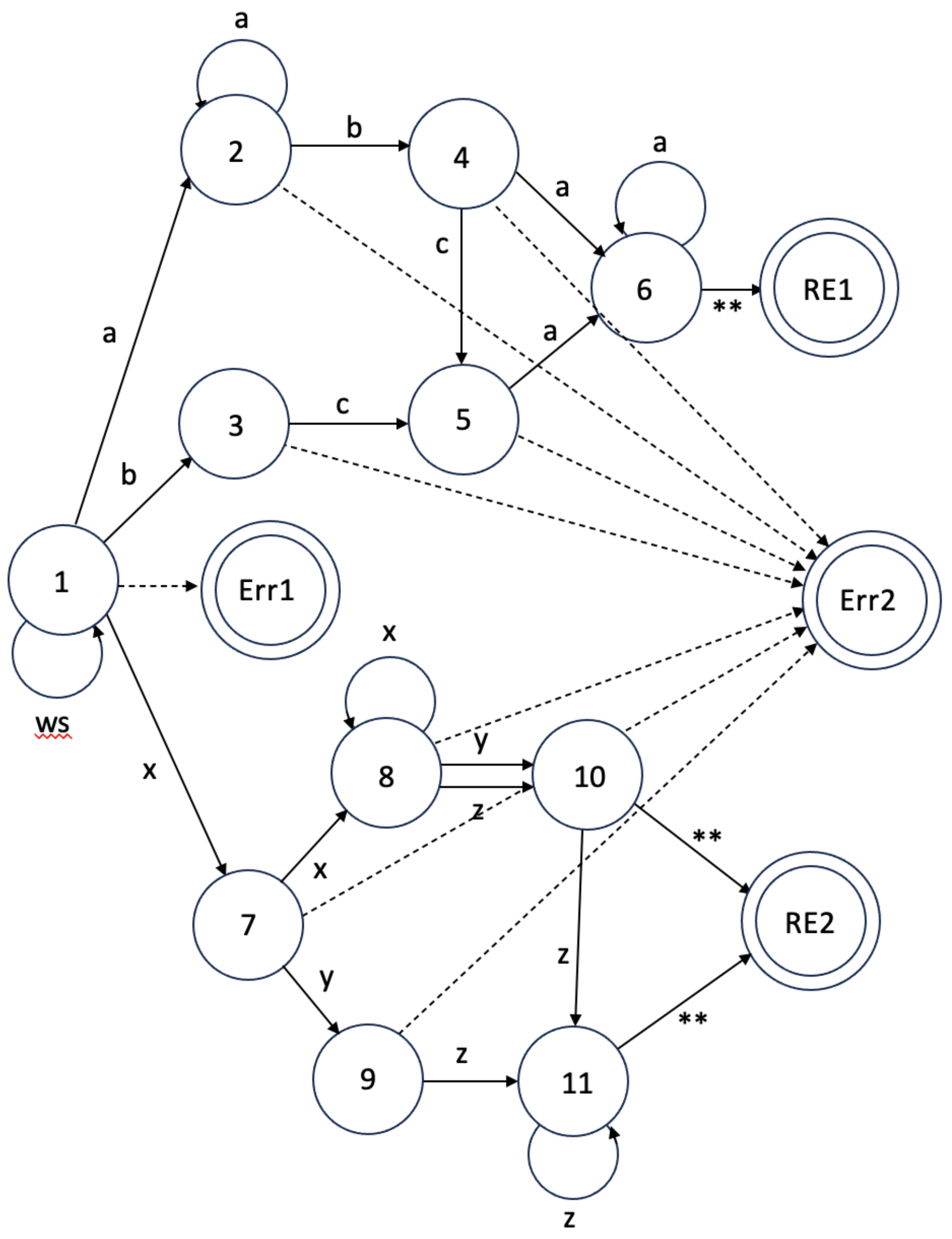
Course Administration

- Exercise 2 Preliminary Exercise
- Project 1 Preliminary Exercise

DFAs as scanners (aka tokenizers)

- Alphabet = {a, b, c, x, y, z, ␣}
- Regular expression 1 (RE1)
  - a* (ab | bc) a+
- Regular expression 2 (RE2)
  - x+ (xy | yz | xz) z*
- Combined
  - (a* (ab | bc) a+) | (x+ (xy | yz | xz) z*)

(a* (ab | bc) a+) | (x+ (xy | yz | xz) z*)



Programming a DFA

- Table

	ws	a	b	c	x	y	z	other
1	1	2	3	Err1	7	Err1	Err1	Err1
2	Err2	2	4	Err2	Err2	Err2	Err2	Err2
3	Err2	Err2	Err2	5	Err2	Err2	Err2	Err2
4	Err2	6	Err2	5	Err2	Err2	Err2	Err2
5	Err2	6	Err2	Err2	Err2	Err2	Err2	Err2
6	RE1	6	RE1	RE1	RE1	RE1	RE1	RE1
7	Err2	Err2	Err2	Err2	8	9	Err2	Err2
8	Err2	Err2	Err2	Err2	8	10	10	Err2
9	Err2	Err2	Err2	Err2	Err2	Err2	11	Err2
10	RE2	RE2	RE2	RE2	RE2	RE2	11	RE2
11	RE2	RE2	RE2	RE2	RE2	RE2	11	RE2

Data Types

- Scalar
- Array
- Record
- Object

Why are we doing this?

What is “Currency”?

- A system of money in general use in a particular country.
- The tangible form of money that is paper bills and coins
- Monetary amount $\rightarrow$ class Money

Class “Money”

- What attributes should it contain?
- What methods should it implement?
- What will it be used for?
  - A problem similar to the one posed for Exercise 1

Interface vs Implementation

- Interface ".h" file
- Implementation ".cpp" file
- Why separate interface and implementation
- Black box concept
  - Programmer needs to know what a method does
  - Programmer needs to know how to use a method
  - Programmer does not need to know the nitty-gritty details of how a method works.
  - For example: vector insert method

Separate compilation

- Money.h
- Money.cpp → Money.o
- makefile

```
Money.o : Money.h Money.cpp
	g++ -c Money.cpp
Exercise2.o : Exercise2.h Exercise2.cpp
	g++ -c Exercise2.cpp
E2.out : Exercise2.o Money.o
	g++ -o E2.out Exercise2.o Money.o
```

Money.h basics

```
#ifndef MONEY_H
#define MONEY_H

#include <iostream>
using namespace std;

class Money
{
    public:
        Money ();
        Money (const Money & M);
        ~Money ();
        Money & operator = (const Money & M);
    private:
        // int dollars, cents;
};

#endif
```

What else?

- Attributes
  - `int dollars, cents;`
- Methods
  - Constructors
  - Mutators (aka setters)
  - Accessors (aka getters)
  - Operators