COSC 385.001 – Automata
Spring 20105
Project # 1

Due: Monday, March 07, 2016, 03:00 PM

Problem - 8

Student: Mateus Meruvia
Instructor: Vojislav Stojkovic

Points:

Problem - 8 - Description

Write a program in your favorite programming language to implement a finite automaton that accepts only (signed and unsigned) integer numbers.

Algorithm

```
//  main.cpp
//  is_identifier
//
//  Created by Mateus Mesturini Meruvia on 2/19/16.
//  Copyright © 2016 Mateus Mesturini Meruvia. All rights reserved.
//

#include <iostream>
#include <string>

bool is_identifier(std::string str){
    int n,i, current=0;
    n = str.length();

    for(i=0; i<n; i++){


        if( (    str[i]>= 65 && str[i]<= 90)
             || (str[i]>= 97 && str[i]<= 122)
             || str[i] == 95 ){ // check if is a letter or underscore using ASCII
table

            current++; // if char =  use this variable to skip the next if
        }

        if (i != 0) { // numbers are not allowed in the first position
            if( str[i]>= 48 && str[i]<= 57 ){ // check if is a number
                if (current == i) { // check if the current position is equal to the
'i' position, if not skips
                    current++;
                }
            }
        }

        if(current == i) { // if 'current' and 'i' have the same value at this point
it means that
            return false;  // none of the previous conditions were fulfilled, so its
not a integer, returns FALSE
        }


    }
    return true;
}

int main(int argc, const char * argv[]) {

    std::string identifier;

    while (1) {

        std::cout << "Enter identifier ";
        std::getline (std::cin,identifier); // reads a string from the standard input


        if(is_identifier(identifier)==true){
            printf("ACCEPTED\n\n");
        }else{
            printf("NOT ACCEPTED\n\n");}
    }
    return 0;
}
```

Explanations

My approach to solve this problem was dividing it in parts. This way I could have a better look at each part individually.

#1 Part - Definition of the language

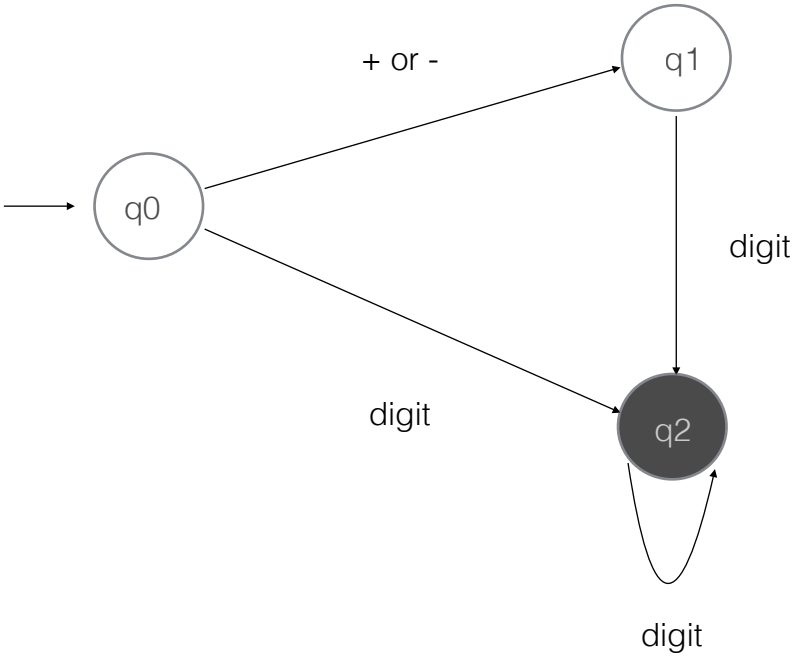
Integers can be easily defined:

1. It has to be a whole number, not a fraction
2. It may start with + or - sign, or not.

#2 Part - Finite Automata

I first drew the finite automata for the proposed problem in a sheet of paper. The automata has only three states, one of which is a finite state (represented as a black circle) because an integer has to end with a digit.

Starting in q0, there are two option: +/- sign or a number. If a +/- sign is read the next state is q1. Once in q1, the only option is going to q2 by reading a digit. Once in q2 multiple digits can be read. q2 is a final state. The second option is starting in q0 with a digit. The next state is q2, which is a final state. Once in q2 multiple digits can be read.



Finite automata for signed or unsigned integers

#2 Part - From automata to C++ coding

In this second part I started coding in C++. My most important insight was that every state becomes an if statement in C++. The full code can be seen in the page 3.

Test Examples

```
Input: 10
Output: ACCEPTED

Input: +10
Output: ACCEPTED

Input: -10
Output: ACCEPTED

Input: 0
Output: ACCEPTED

Input: 10+
Output: NOT ACCEPTED

Input: 10-
Output: NOT ACCEPTED

Input: 1+0
Output: NOT ACCEPTED

Input: 1-0
Output: NOT ACCEPTED

Input: a10
Output: NOT ACCEPTED

Input: 10a
Output: NOT ACCEPTED

Input: 1.2
Output: NOT ACCEPTED

Input: test
Output: NOT ACCEPTED
```