Systems Software
Assignment 2: Strings, File IO, and command-line arguments

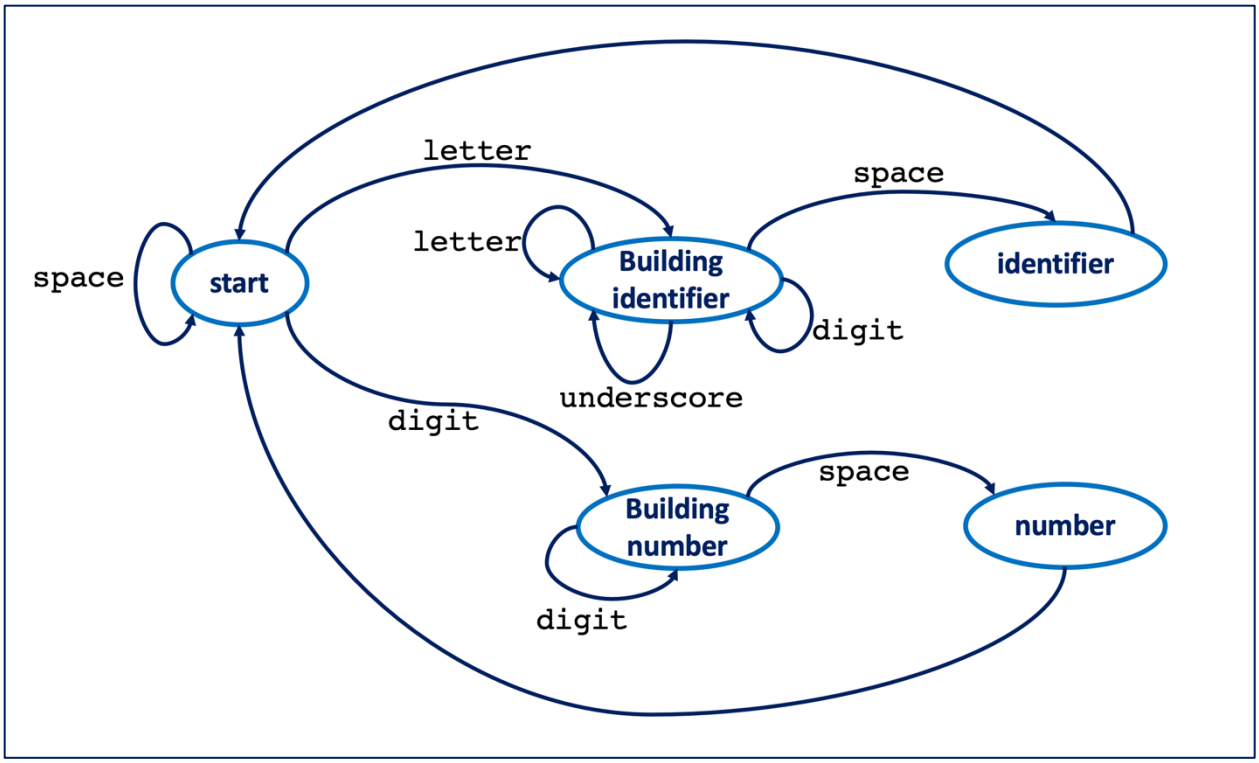
Assignment Objectives

Students should demonstrate the following abilities:

1. Access text files for reading and writing from C programs
2. Use C string library functions to manipulate text
3. Pass command-line arguments to C programs

Assignment

A finite state machine (FSM) consists of a set of states, a set of transitions, and inputs/outputs. It is used to model the behavior of machines or programs where the current state is determined by the previous state and the values of the inputs. In the FSM below, the ovals represent the states and the connections between the ovals represent the transition from one state to another. The labels (text) over the transitions represent the input value that triggers the transition. The FSM shows how an input text is processed to identify data items as identifier or number. Such a machine can be used by a lexical analyzer, a piece of software that is part of a compiler, to find the list of symbols used in a C program for example.



The FSM starts at the state START and will remain in the same state as long as the input character is a space (blank or new line). The machine will transit from START to the state BUILDING_NUMBER when the input character is a digit. It will remain in the same state as long as the next character is a digit and will transit to the state NUMBER when a space is read. The state NUMBER means that a data item of type number has been read and the machine will transit to the state START to start the identification of a new data item. In the same way, you can describe the behavior of the machine for the other states.

Write a program that uses an enumerated type to represent the names of the states. Your program should process an input file and identify each data item as identifier, keyword, or number. A keyword is an identifier that is found in the list of keywords. The list of keywords in C language is provided at the end this assignment for your reference.

Here is a sample input and output of your program:

Input:

```
include
int main
  int money 220 fifty 0 twenty 0 ten 0
  printf
  money  get_input
  find_num_bills money  fifty twenty ten
```

Output:

```
include - identifer
int - keyword
main - identifer
int - keyword
money — identifier
220 - number
fifty — identifier
0 - number
twenty — identifier
0 - number
ten — identifier
0 - number
printf - identifer
money - identifer
get_input - identifer
find_num_bills - identifer
money - identifer
fifty - identifer
twenty - identifer
ten — identifier
Number of identifiers: 14
Number of keywords: 2
Number of numbers: 4
```

Define a function **`transition`** that accepts two inputs, the current state of the state machine and the current input character and returns the value of the next state of the machine. The method should implement the behavior of the FSM as shown in the diagram above. The main method calls **`transition`** for each character read from the input file to update the current state with the state returned by **`transition`**. Below is a code snippet from the main function to show you how to use the method **`transition`**.

```
typedef enum{ START, IDENTIFIER, NUMBER,
              BUILDING_IDENTIFIER, BUILDING_NUMBER
} state;

state current_state = START;
char current_char = read one character from the input file
while(current_char != EOF){
  current_state = transition(current_state, current_char);
  if(current_state == IDENTIFIER){
    check if the identifier is a keyword
    display the data item with the word identifier or keyword
    current_state = START;// move to new data item
  }
  else if (current_state == NUMBER){
    display the data item with the word number
    current_state = START;
  }
  current_char = read the next character from the input file
}
```

Include the header file **`<ctype.h>`** to use the library functions such as `isdigit(ch)`, `isalpha(ch)`, and `isspace(ch)` that return 1 if their character argument `ch` is a digit, a letter or a space (`' '` or `'\n'`) respectively.

Save your C program in a file named **`symbol_analyzer.c`** . Your program should accept one command-line argument that is the name of the input file. Test your program using the input file **`sample.txt`** provided with this assignment. Your program should display the list of identifiers, keywords, and numbers in addition to the following statistics for **`sample.txt`**.

```
Number of identifiers: 54
Number of keywords: 19
Number of numbers: 11
```

Submit the file **`symbol_analyzer.c`** on courseSite.

List of C language keywords:

```
auto        break       case        char        const       continue
default     do          double      else        enum        extern
float       for         goto        if          int         long
register    return      short       signed      sizeof      static
struct      switch      typedef     union       unsigned    void
volatile    while
```