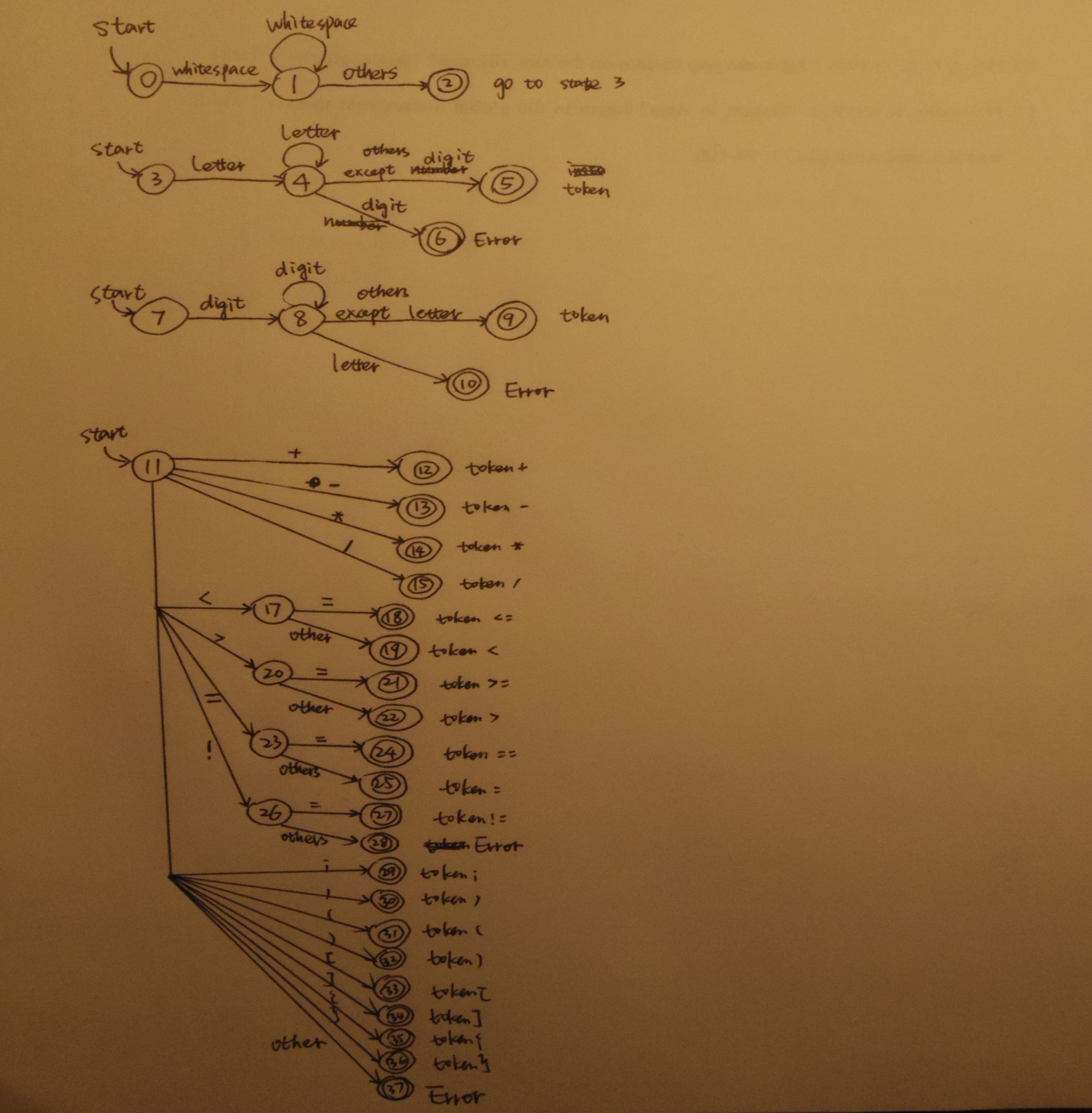
Program1

1. Transition Diagram



1. Pseudo code

Void main()

{

Call pre() to delete the comments

Call lex() to deal with the code

Create a file called Keywords.txt

Out put the array Keywords to the file

Create a file called Identifiers.txt

Out put the array identifiers to the file

Create a file called Numbers.txt

Out put the array numbers to the file

Create a file called Tokens.txt

Out put the array tokens to the file

}

String pre()

{

Read from the file called code.c

Put it in a string

Traverse all the chars in the string

if the char is / and the next is \*

while the char is not \* or( the char is \* and the next is /)

The index increase by 1

Put the char into an array called targetcode

Put targetcode into a string called fileContent

}

Bollean isKeyword()

{

Traverse the array called Keyword

Check if the word is keyword and return the result

}

Token install\_id()

{

Create a string called iden to store the word read from the targetcode

Call method iskeyword to check if it is a keyword

If so, call method install\_key

Else, check if it has existed in the array Identifiers

If so, break

Else, add the identifier into the array Identifiers

Create a token with the type id and the index and return it

}

Token install\_key()

{

Create a string called keyw to store the word read from the targetcode

Check if the word has existed in the array Keywords

If so, break

Else, add the keyword into the array Keyword

Create a token with the type key and the index and return it

}

Token install\_num()

{

Create a string called numb to store the number read from the targetcode

Check if the number has existed in the array Numbers

If so, break

Else, add the number into the array Numbers

Create a token with the type num and the index and return it

}

Token install\_symbol()

{

Create a string called numb to store the special symbol read from the targetcode

Check if the symbol has existed in the array SpSymbols

If so, break

Else, add the symbol into the array SpSymbols

Create a token with the type sp and the index and return it

}

Char nextchar()

{

Get the next char from the array targetcode

The index increase by 1

}

boolean isletter()

{

Check if the char is a letter

}

boolean isdigit()

{

Check if the char is a digit

}

Token[] lex

{

Call nexttoken()

Add the return value to the array TokenTable

}

Token nexttoken()

{

While(true)

{

Swich(state)

{

Case -1:

Return null to end

Case 0:

If the char is whitespace

Change the state to 1

Else

Change the state to 3

Break

Case 1:

If the char is whitespace

Change the state to 1

Else

Change the state to 3

Break

Case 3:

If the char is letter, change the state to 4

Else, change the state to 7

Break

Case 4:

If the char is letter, change the state to 4

Else, check if the chat is digit, if so, print: No digit after letter!

Else, change the state to 5 and call install\_id

Break

Case 7:

If the char is digit, change the state to 8

Else, change the state to 11

Break

Case 8:

If the char is digit, change the state to 8

Else, check if the char isletter, if so, print: No letter after digit!

Else, change the state to 9 and call install\_num

Break

Case 11:

If the char is +

Change the state to 12

Case 12:

Call install\_symbol(“+”)

break

......check all the special symbols just like +

Default:

Print: Error:Invalid symbol!

Return null

}

}

}

1. Error reporting algorithm summary

Check every char in the code

If one char is letter and the next char is digit

Print : Error! No digit after letter!

If one char is digit and the next char is letter

Print: Error! No letter after digit!

If one char is ! and the next char is not =, print: Error! No symbols after !!

If all the cases are checked and is not accepted, print: Invalid symbol!

1. Output