**과목명: 시스템프로그래밍**

**홀수**

**<<Project #1>>**

**서강대학교 컴퓨터공학과**

**20181593**

**계인혜**

목 차

1. **프로그램 개요**
2. **프로그램 설명**
   1. 프로그램 흐름도
3. **모듈 정의**
   1. 정의한 모듈에 대해 간략적인 설명
4. **전역 변수 정의**
5. **코드 설명**

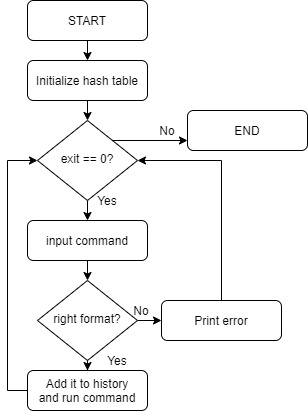
**6 테스트 케이스**

**1. 프로그램 개요**

SIC/XE 머신을 구현하기 위한 전 단계로서 셸(shell)과 컴파일을 통해서 만들어진 object코드가 적재되고 실행될 메모리공간과 mnemonic을 opcode 값으로 변환하는 opcode table과 관련 명령어들을 구현하는 프로그램이다.

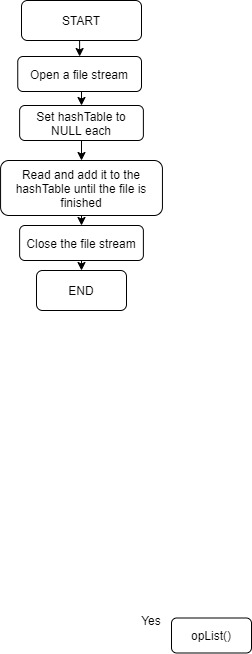
**2. 프로그램 설명**

2.1 프로그램 흐름도

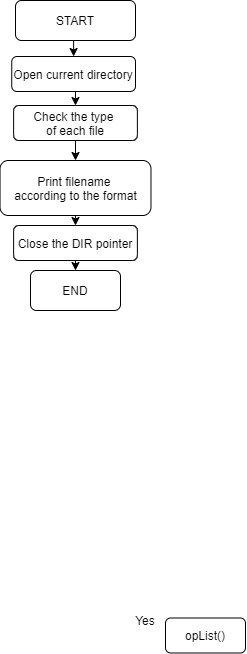
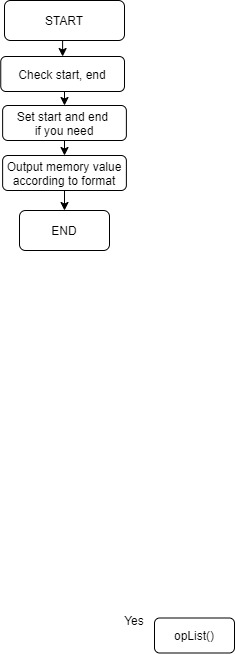


2.2 세부 프로그램 흐름도

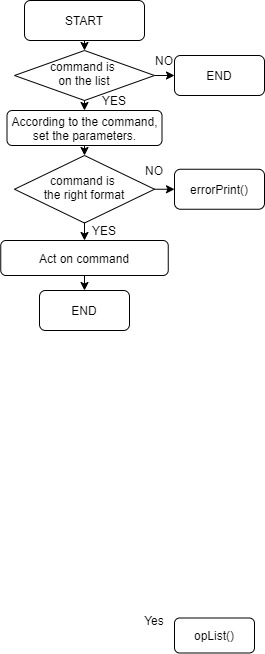
① initialize -> hash table을 초기화하는 함수



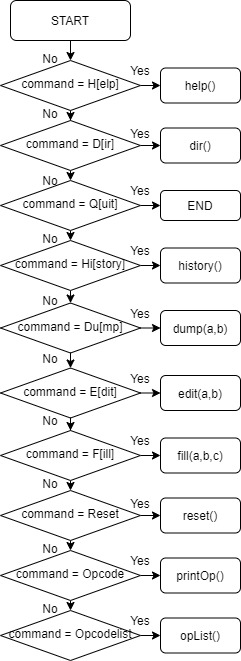
② dir-> 명령어 dir을 처리하기 위한 함수 ③ dump -> 명령어 dump를 처리하기 위한 함수

④ makeCommend -> 명령어가 list에 있는지 확인하고 명령어에 따라 parameter를 나누는 함수



⑤ run -> 명령어에 따라 수행해야 하는 함수를 호출하는 함수



**3. 모듈 정의**

3.1 모듈 정의

① 20181593.c

|  |
| --- |
| 이 모듈은 메인 함수가 있는 모듈로 사용자로부터 한 줄씩 명령어를 입력받아 이를  실행할지 말지를 결정한다. 명령어를 실행하게 되면 history에 해당 명령어를 추가하고,  명령어를 실행하지 않으면 해당하는 경우에 맞는 에러 메세지를 출력한다. 사용자가 q  를 입력하여 프로그램을 끝내기 전에 생성했던 해쉬 테이블과 history를 저장한 linked  list를 삭제한다. |

② shell.c

|  |
| --- |
| 이 모듈은 help, dir, history 명령어를 처리하기 위한 함수가 들어있는 모듈이다.  history linked list를 구현하기 위하여 가장 끝 부분에 입력받은 명령어를 삽입하는  addHistory, 프로그램이 종료할때 history linked list를 삭제하기 위한 removeHistory  함수를 따로 구현하였다. |

③ opcodeTable.c

|  |
| --- |
| 이 모듈은 opcode mnemonic, opcodelist 명령어를 처리하기 위한 함수가 들어있는  모듈이다. 해시테이블을 초기화하는 함수(initialize), 프로그램이 종료될 때 생성된 해시  테이블을 삭제하는 함수(removeTable), 해시테이블에 새로운 mnemonic을 추가하는 함  수(add), mnemonic에 따라 다른 해시 값을 생성하는 함수(getHash), mnemonic을 사용하  여 입력받은 mnemonic의 opcode를 찾아낼 수 있게 하는 함수(getOp), opcode를 출력  하는 함수(printOp), opcodelist를 차례대로 출력하는 함수(opList)가 있다. |

④ memory.c

|  |
| --- |
| 이 모듈은 dump, edit, fill, reset을 처리하기 위한 함수가 들어있는 모듈이다. |

**4. 전역 변수 정의**

① 20181593.c

|  |
| --- |
| int command : enum을 사용하여 명령어들을 구분한 것  char inputCommand[] : 사용자가 입력한 명령을 저장  char token[][] : 띄어쓰기와 콤마를 기준으로 명령을 잘라서 저장 |

② shell.c

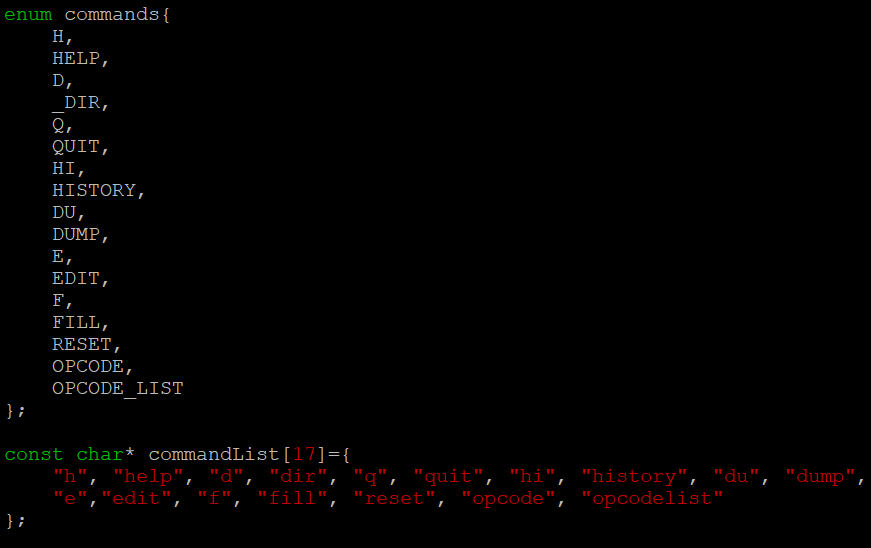
|  |
| --- |
| node\* head, tail : history linked list의 맨 앞과 맨 끝 부분을 나타내는 노드  int cnt : history linked list의 노드의 개수 |

③ opcodeTable.c

|  |
| --- |
| hashNode\* hastTable[SIZE] : SIZE보다 훨씬 많은 mnemonic이 들어올 때 생기는 충돌을 최소한으로 하기위하여 linked list로 구현된 해쉬 테이블  hashNode opTable[0xFF] : 각각의mnemonic에 해당하는 정보를 저장 |

④ memory.c

|  |
| --- |
| int memory[MAX] : 사이즈가 1MB인 가상의 메모리 공간  int last : dump의 실행으로 출력된 마지막 address를 저장 |

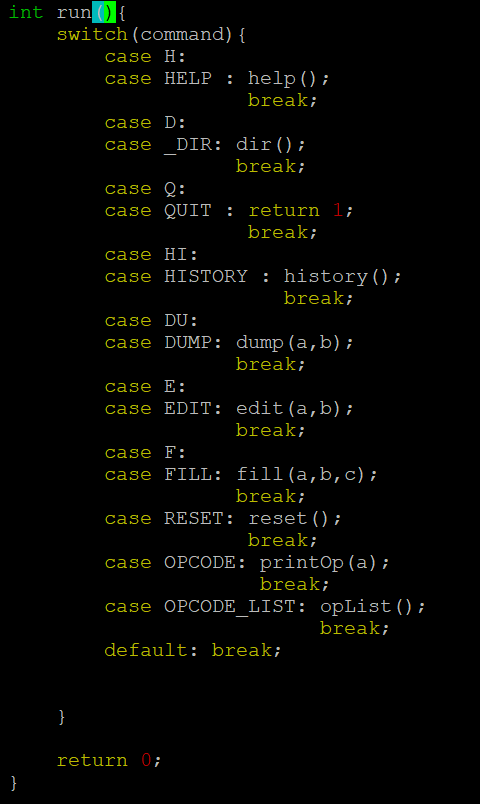


**5. 코드 설명**

① 20181593.c

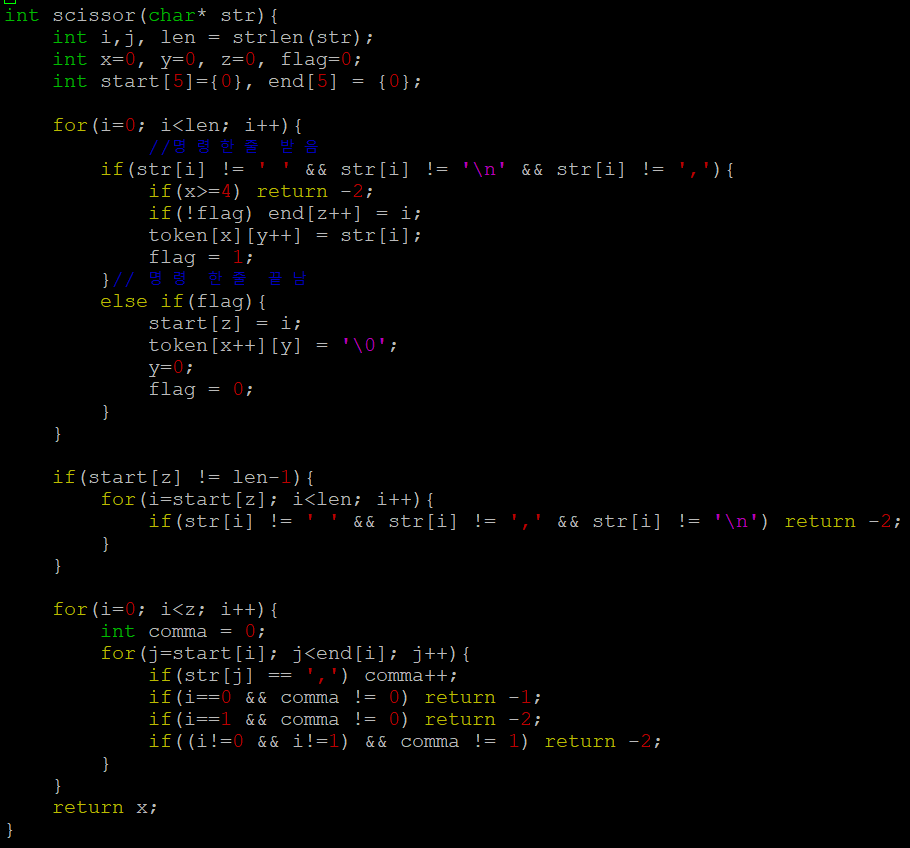
Ⓐ enum

소스의 가독성을 높이기 위해 각각의 명령어를 enum을 사용하여 처리하였다. 이때, DIR이라는 명령어는 이미 리눅스 상에서 키워드로 정의되어 있기 때문에 '\_'를 추가하여 \_DIR이라고 설정했다.

Ⓑ run

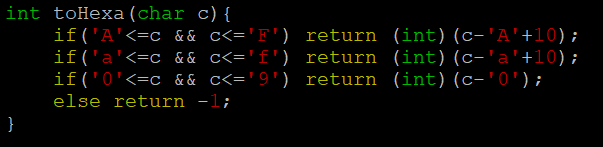
이전에 정의해둔 enum을 사용하여 입력받은 명령에 따른 함수를 호출한다. 정의되지 않은 명령어가 들어오면 어떤 함수도 호출하지 않고 함수를 빠져나간다. enum을 사용하여 코드의 가독성을 높였다.

Ⓒ scissor



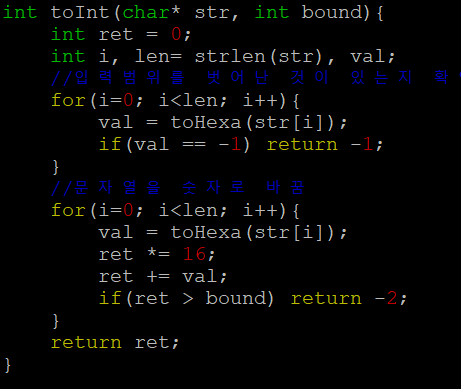
해당 함수는 사용자로부터 받은 명령어를 형식에 맞춰 저장한다. 우선, 첫 번째 for문은 명령어를 공백과 콤마 엔터를 기준으로 나누어 token에 저장한다. 이때 들어올 수 있는 단어의 최대 개수는 4개로(fill a,b,c) 단어의 개수가 이를 초과하면 예외 처리를 한다. 다음 for문에서는 콤마의 개수를 세고 형식에 맞지 않는 명령어에 대한 예외 처리를 한다. 리턴 값은 잘린 단어의 개수이다.

Ⓓ toHexa



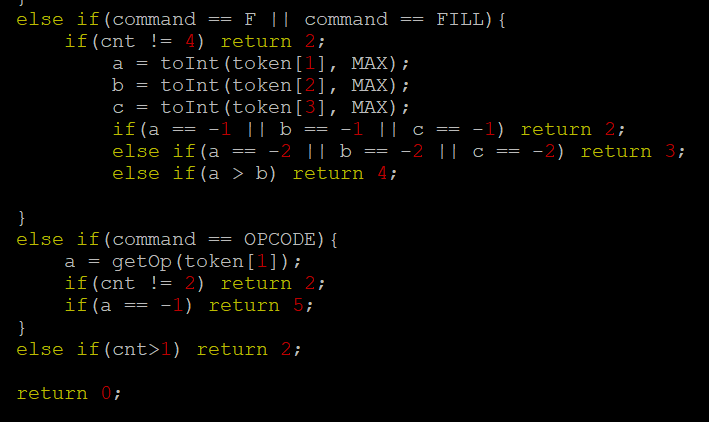
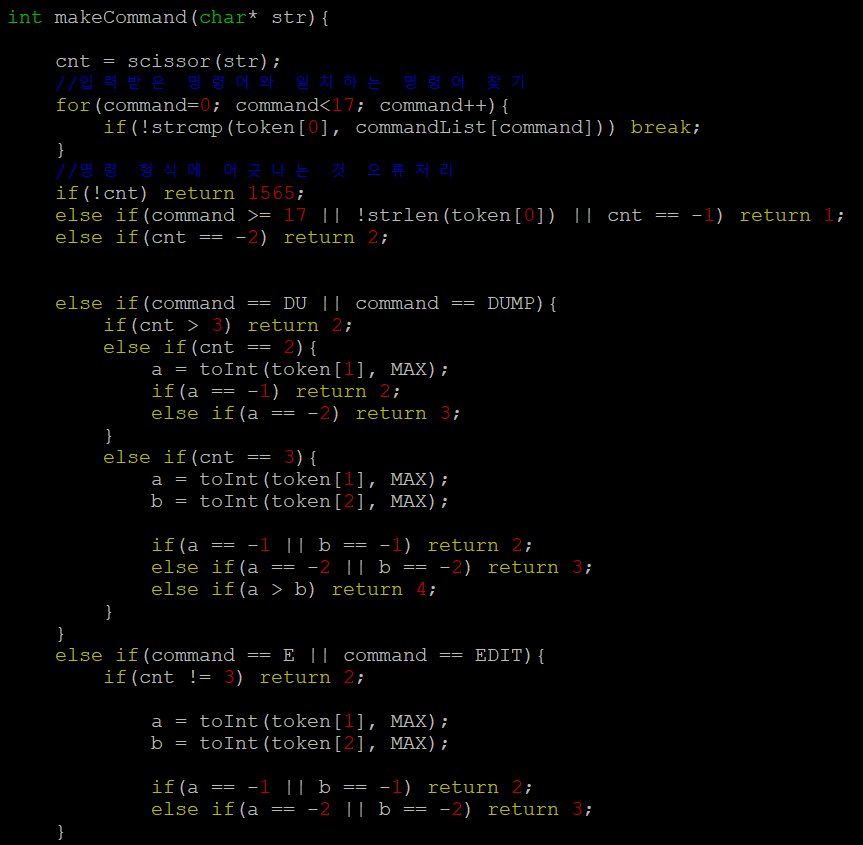
이 함수는 character c를 16진수로 바꾸는 것으로 입력 값이 16진수의 범위를 벗어나는 경우에는 -1을 리턴하고 종료한다.

Ⓔ toInt

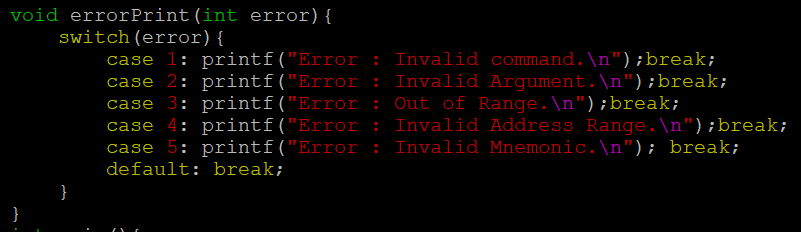


이 함수는 사용자가 입력한 문자열 중 16진수를 찾아 이를 올바른 정수 값으로 저장하는 함수이다. 16진수가 아닌 character가 나오면 -1을 리턴하고 16진수의 범위를 벗어나는 경우 -2을 리턴한다.

Ⓕ makeCommand

이 함수는 사용자가 입력한 명령과 commandList에 저장된 명령의 일치 여부를 판단하고 해당하는 명령어에 따라 parameter를 설정하고 명령어의 형식을 검사하는 함수이다. Scissor에서 잘라 놓은 명령어들을 순서대로 parameter a,b,c에 저장한다. 

Ⓖ errorPrint



위 함수는 형식에 어긋나는 사용자의 input command를 유형별로 분류하여 그에 맞는 에러 메세지를 출력하는 함수이다.

Ⓗ main

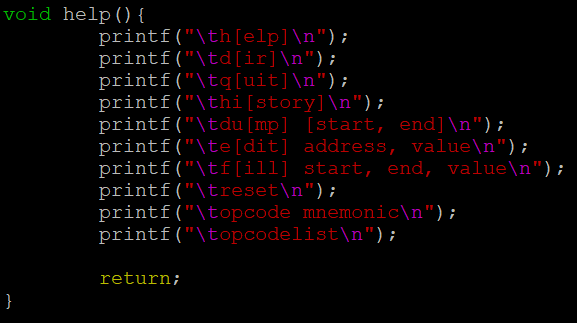


위 함수는 메인함수로 프로그램이 시작할 때 해쉬테이블을 생성하며(initiate), 명령어 'q'를 입력받고 프로그램을 종료할 때는 생성했던 해쉬테이블과 history linked list를 모두 삭제한다. 'q[uit]' 명령어를 입력받기 전에는 사용자로부터 명령어를 한 줄씩 입력받고 이를 처리하는 과정을 반복한다. 한 줄씩 입력받을 때마다 전역변수를 초기화하며, 입력의 마지막을 항상 '\n'으로 설정한다. 명령어를 처리하는 과정(makeCommand)에서 에러가 발생하면 에러를 출력하고(errorPrint), 그렇지 않으면 history linked list에 명령어를 삽입하고 이를 수행한다.

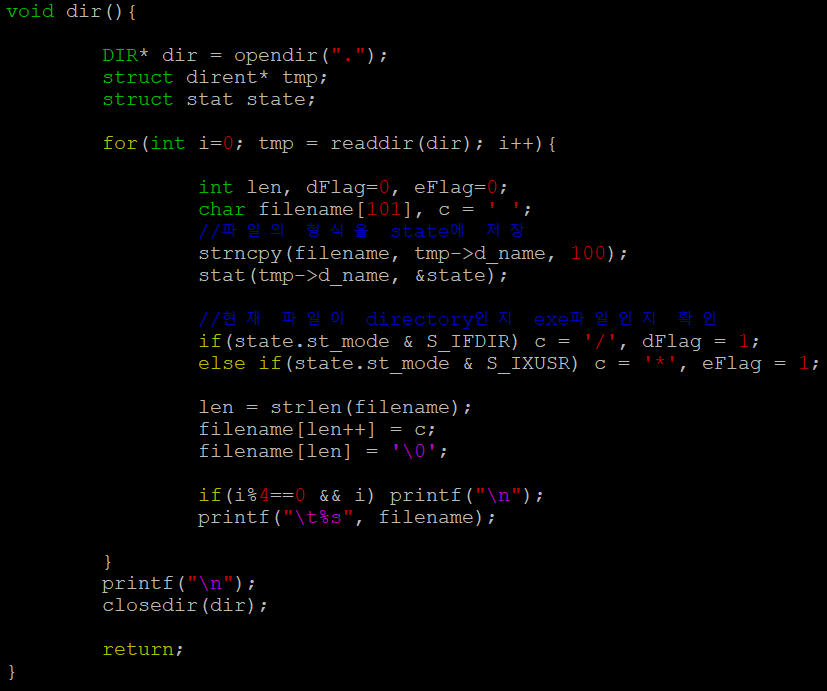
② shell.c

Ⓐ help

위 함수는 shell에서 실행 가능한 모든 명령어들의 리스트를 출력한다.



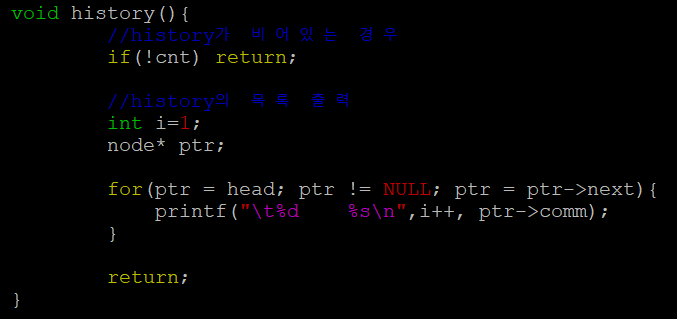
Ⓑ dir



위 함수는 현재 디렉터리를 열어 그 안에 있는 파일들을 형식에 맞춰 출력하는 함수이다. 이때 실행 파일은 이름 옆에 '\*' 표시를, 디렉터리는 이름 옆에 '/'를 붙인다.

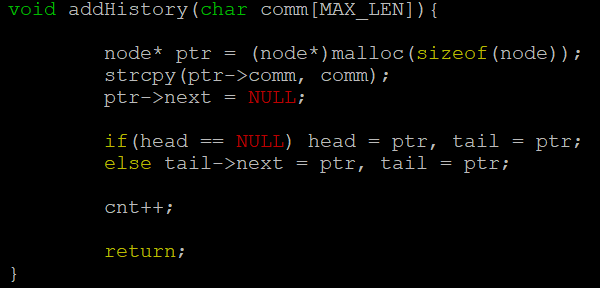
현재 디렉터리 안에 있는 모든 파일을 살펴서 그 파일이 실행 파일인지 디렉터리인지 확인하고 만약 그렇다면 flag를 켠다. 한 줄에 4개의 파일명을 출력하며 함수가 끝나기 전에는 열었던 디렉터리를 닫는다.

Ⓒ history

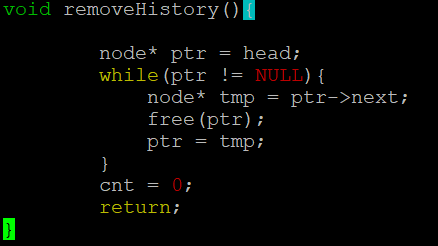


위 함수는 history 명렁어를 처리하기 위한 함수로 history linked list에 저장된 내용을 차례대로 출력한다.

Ⓓ addHistory

 이 함수는 올바른 명령이 들어왔을 때 그 명령을 history linked list에 추가하는 함수이다. linked list가 비어있는 경우와 그렇지 않은 경우로 나누어서 함수를 구현하였으며, 추가 후에는 linked list에 들어있는 노드 수를 나타내는 cnt를 1 증가시키고 함수를 종료한다.

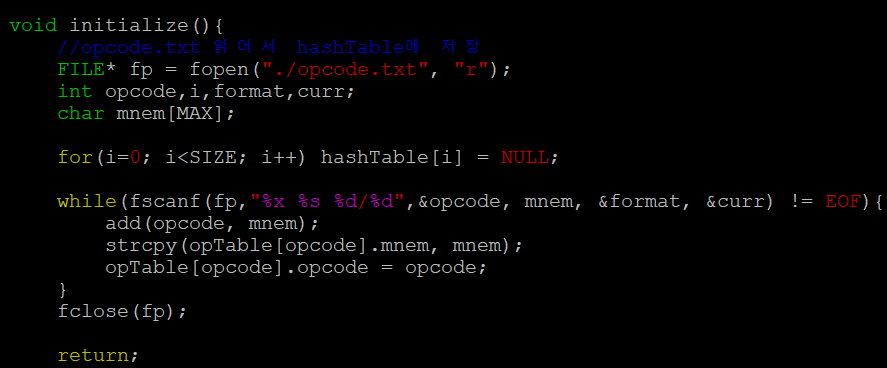
Ⓔ removeHistory



이 함수는 프로그램이 종료되기 전에 지금까지 사용자로부터 입력받은 명령어들을 저장해 놓은 history linked list를 전부 free해주는 함수이다.

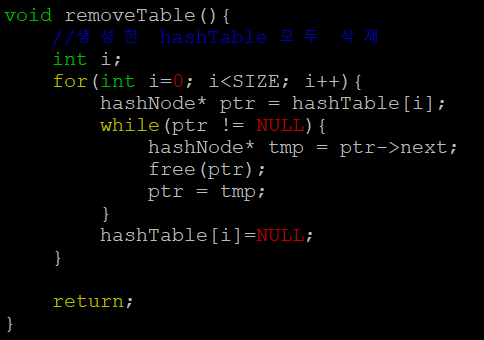
③ opcodeTable.c

Ⓐ initialize



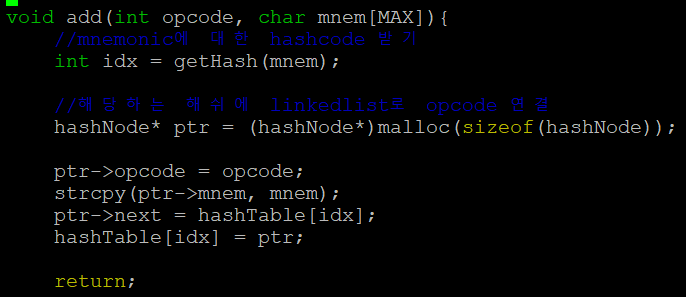
위 함수는 opcode.txt를 읽어 각 mnemonic에 해당하는 opcode와 format을 저장한다. 각각의 opcode를 크기가 20인 hashTable에 저장한다. 이때 hashTable의 크기에 비해 테이블에 넣어야 할 mnemonic의 개수가 훨씬 많아 충돌이 생길 위험이 있다. 이를 방지하기 위해 linked list 형태로 구현하였다.

Ⓑ removeTable



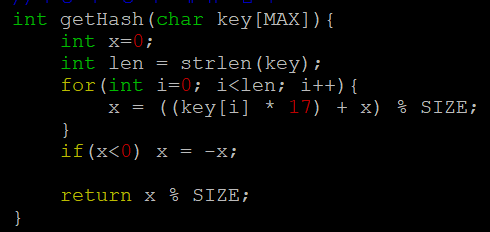
이 함수는 프로그램이 종료되기 직전에 생성해 두었던 hashTable을 전부 free하는 함수이다.

Ⓒ add

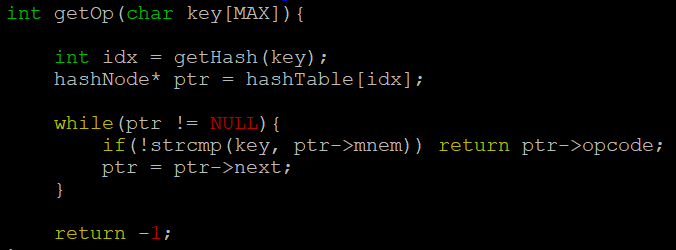


이 함수는 mnemonic에 따라 hash code를 생성하여 linked list 형태로 구현한 hashTable에 삽입하는 함수이다.

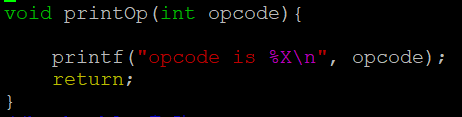
Ⓓ getHash

 이 함수는 hash function으로 mnemonic에 따라 hash 값을 생성하는 함수이다. polynomial rolling hash function으로 구현하였으며 x는 항상 양수이다. 이때 modulo에 사용되는 양수는 대체로 큰 소수 값으로 설정하는 것이 좋다. 그러나 해시 테이블의 크기가 20이기 때문에 여기서는 그 값을 20으로 설정하였다.

Ⓔ getOp

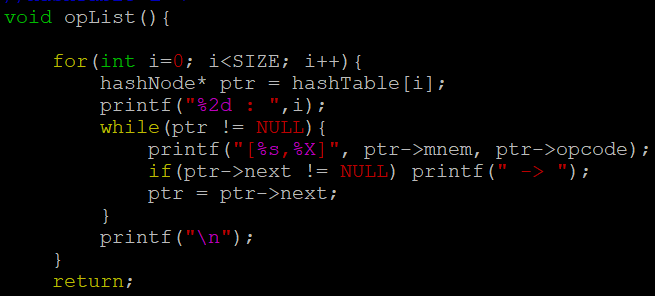
 이 함수는 opcode mnemonic 명령어를 처리하기 위한 함수로 mnemonic을 입력받으면 해시 값을 이용하여 hashTable linked list와 비교 후 명령어에 해당하는 opcode를 출력한다.

Ⓕ printOp



이 함수는 opcode를 parameter로 받아서 출력하는 함수이다.

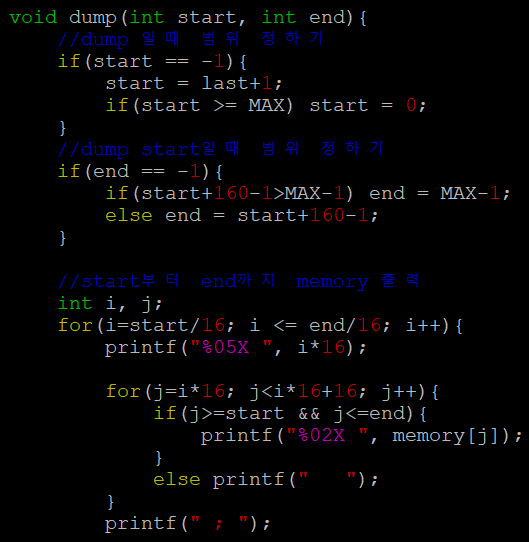
Ⓖ opList



이 함수는 opcodelist 명령어를 처리하기 위한 함수로 크기가 20인 opcode hashTable을 형식에 맞춰 차례로 모두 출력한다.

④ memory.c

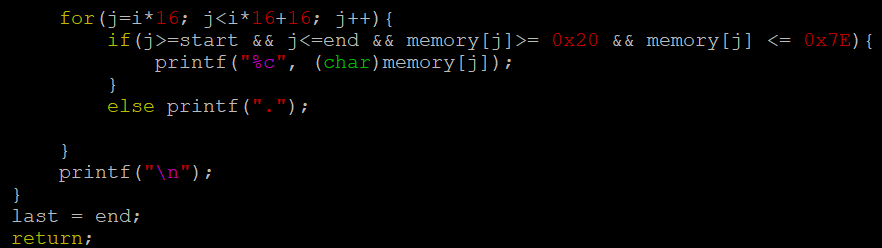
Ⓐ dump



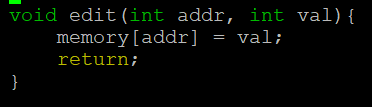
이 함수는 dump 명령어를 처리하기 위한 함수로 세가지 포맷을 처리한다.

만약 입력된 명령어가 dump라면 start에 이전에 출력한 마지막 주소 + 1 번지를 assign 하고, end에는 start부터 10라인을 더한 값을 넣는다. 입력된 명령어가 dump start라면, end에는 start부터 10라인을 더한 값을 넣는다. 만약 end 값이 메모리 범위를 벗어나는 경우 맨 마지막 번지로 설정한다. 마지막으로 dump start, end로 명령이 들어온 경우는 입력값이 범위를 벗어나지 않는 이상 그대로 유지한다.

start, end 값을 설정한 후에는 1라인(16바이트)씩 형식에 맞춰 출력한다. 이때, ASCII 코드로 출력해야 하는 범위를 벗어난 값은 '.'으로 출력한다.

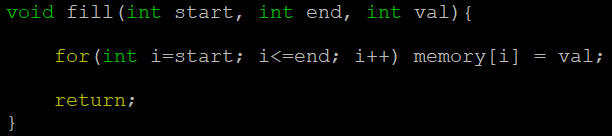


Ⓑ edit



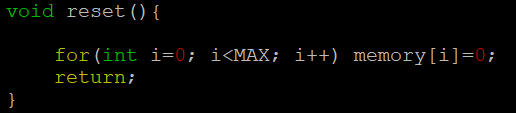
이 함수는 edit 명령어를 처리하기 위한 함수이다.

Ⓒ fill



이 함수는 fill 명령어를 처리하기 위한 함수이다.

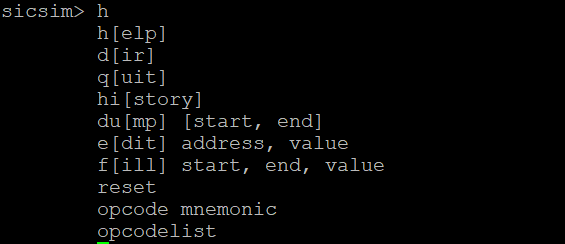
Ⓓ reset



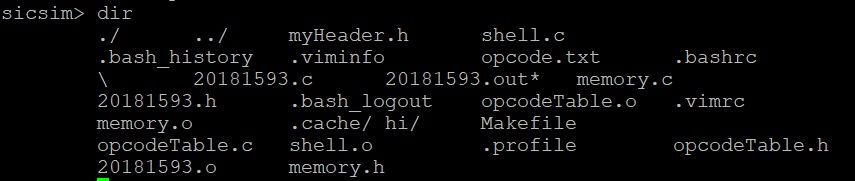
이 함수는 reset 명령어를 처리하기 위한 함수이다.

**6. Testcase**

① h[elp]



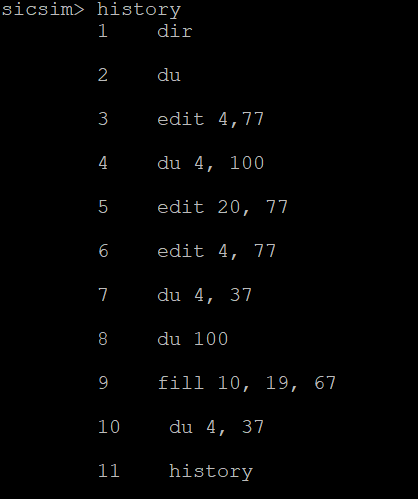
② d[ir]



③ q[uit]



④ hi[story]

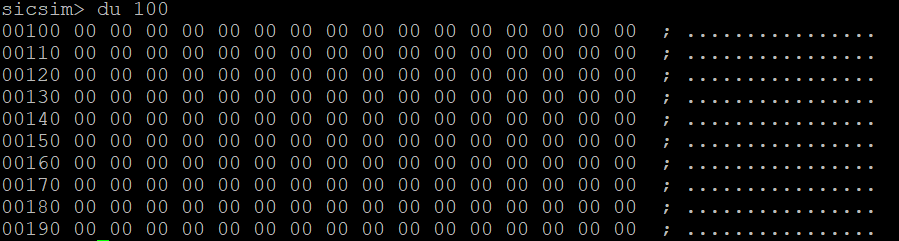


⑤ du[mp]

<du>

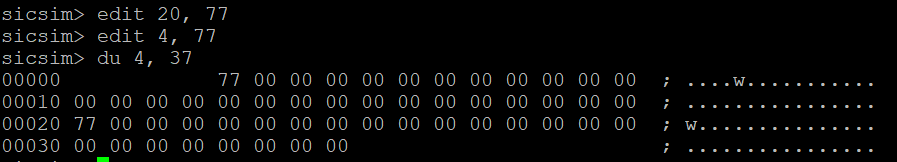


<du 100>



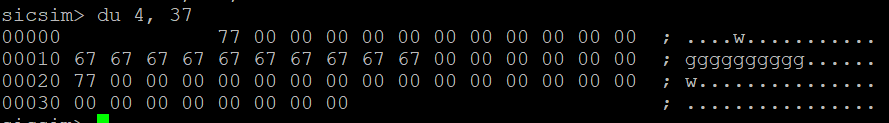
⑥ e[dit]

<edit 20, 77> <edit 4, 77>



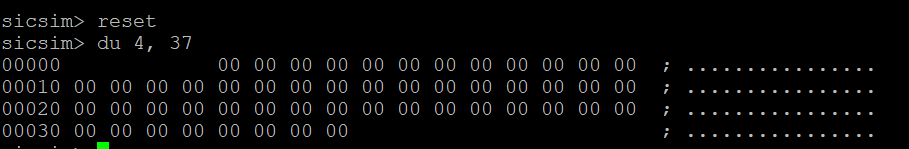
⑦ f[ill]

<fill 10, 20, 67>



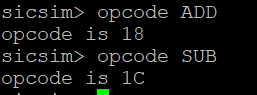
⑧ reset

<reset>



⑨ opcode mnemonic

<opcode ADD> <opcode SUB>



⑩ opcodelist

<opcodelist>

