**프로젝트 보고서**

2017270967 심다송

1. 프로젝트 개요
   1. 주제 선정 동기
      1. 참신성

게임 같은 경우에는 이미 오픈 소스가 많기 때문에 참신하지 않다고 생각

컴퓨터구조를 학습하기 위한 실행 프로그램(윈도우 프로그램)을 찾아 본 결과, 찾지 못하였기 때문에 참신하다고 생각

* + 1. 라인 수

약 1,300줄로써, 최소 라인인 1,000줄 이상 => 조건 만족

* + 1. 실용성 중시

본인의 공부에도 도움이 될 뿐만 아니라 github에 오픈 소스로 올려서 다른 사람들에게도 도움이 되도록 하기 위함

* 1. 프로젝트 목표
     1. 어셈블리 언어를 입력 받으면 기계어로 바꿔주는 프로그램

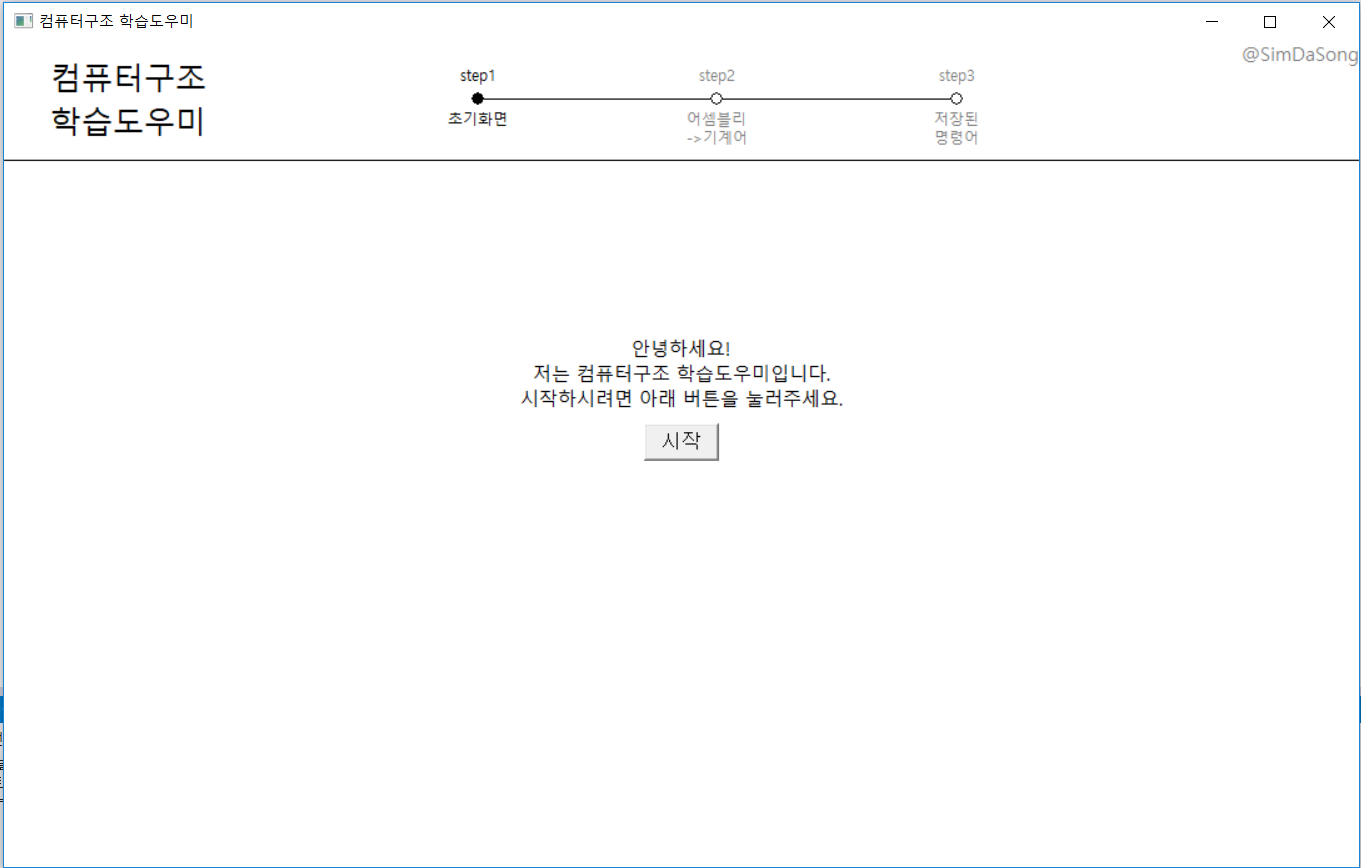
어렵지만 프로그래머에게 도움이 되는, 컴퓨터 구조 과목의 어셈블리언어의 기계언어로의 변화에 대한 이해를 돕기 위함

* + 1. UI 우선시

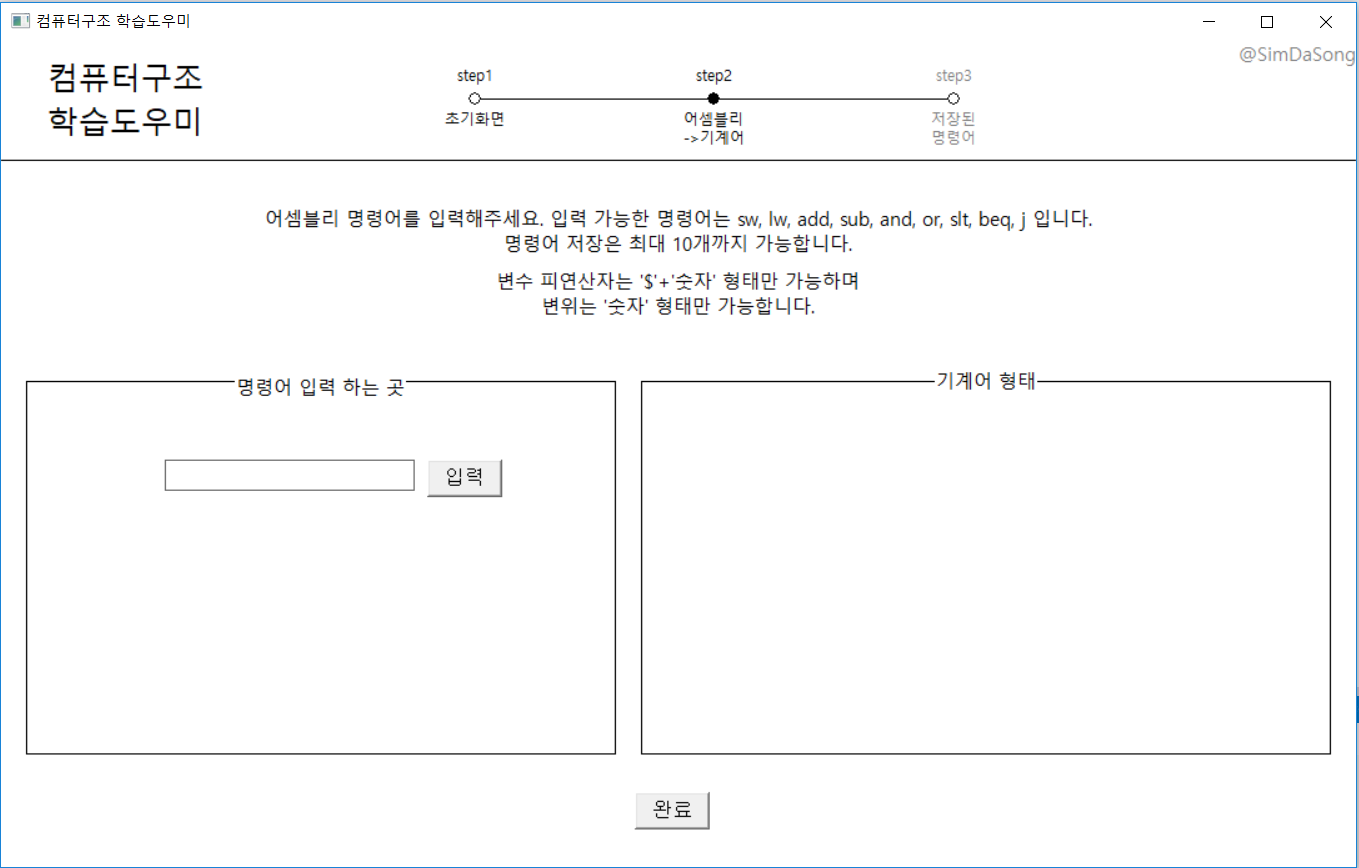
콘솔 프로그램이 아닌 윈도우 프로그램으로 만드는 이유는 UI를 위함이라 생각

UI (글꼴, 반응형 화면 등)을 우선시 한 프로그램을 작성

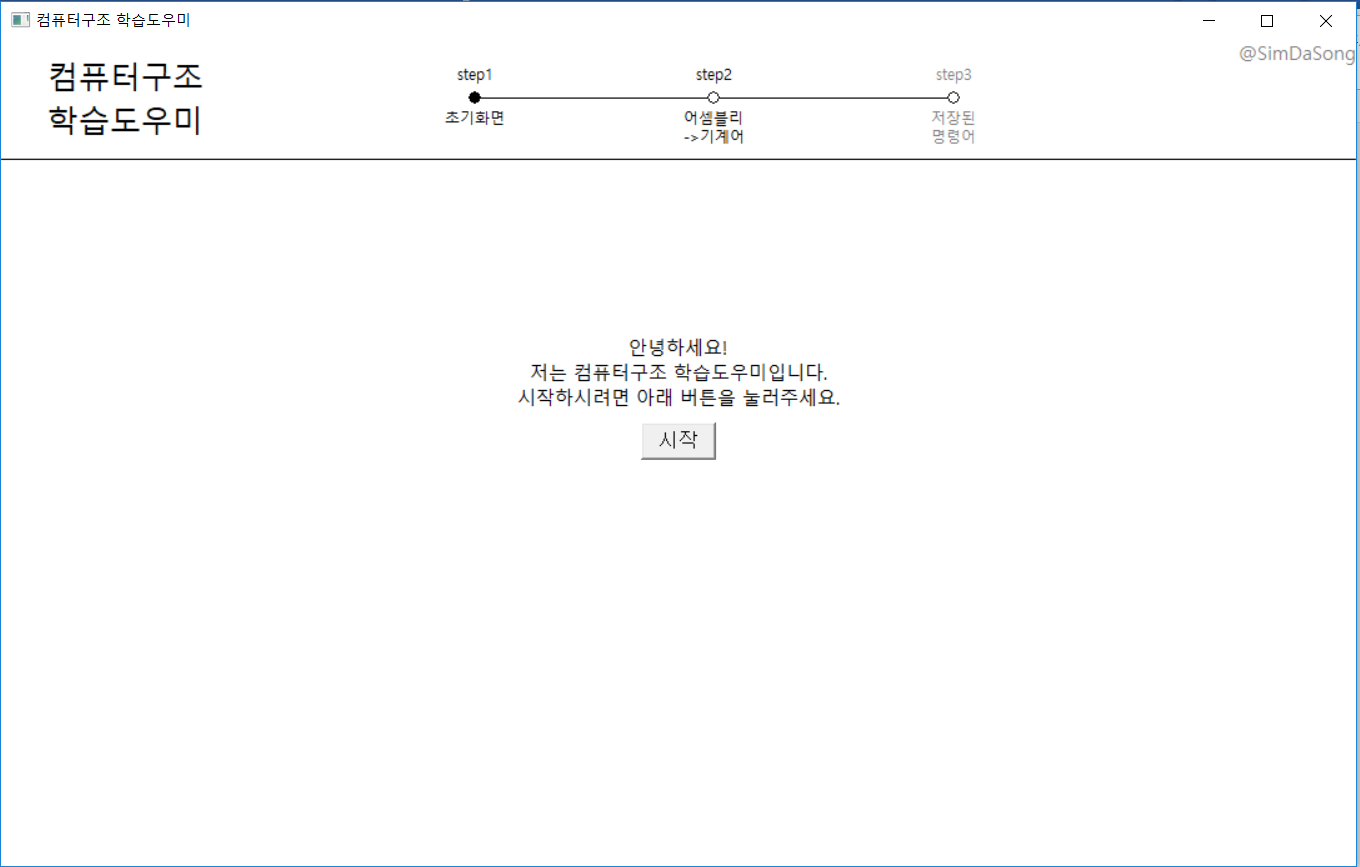
1. 프로그램 계획(단계)
   1. 시작 버튼을 누르면 어셈블리 명령어를 입력할 수 있는 화면으로 넘어감
   2. 어셈블리 언어로 된 명령어를 edit control에 입력 받음
   3. 입력 받은 명령어는 바로 기계어 형태로 보여주고, 해당 명령어를 저장할 수 있음.
   4. 명령어 저장은 최대 10개까지 가능. 10개 넘으면 경고 메시지 뜸
   5. 입력 받은 명령어가 명령어 형식에 맞지 않다 => 경고 메시지 뜸
   6. 완료 버튼을 누르면 다음 화면으로 넘어감
   7. 이때, 콤보 박스 리스트에 있는 명령어들을 instructions 배열에 저장
   8. 입력한 명령어가 없다면 경고 메시지 뜸
   9. 다음 화면에서 저장된 명령어들을 기계어로 변환하여 화면에 출력
2. 설계 및 구현
   1. 설계 개념 및 동작 원리
      1. 각 명령어의 Opcode를 int형에 10진수로, TCHAR 배열(문자열)에 2진수 정보를 담고 있는 Opcode 구조체 선언 => 가독성을 위해
      2. 각 명령어의 Funct를 int형에 10진수로, TCHAR 배열(문자열)에 2진수 정보를 담고 있는 Funct 구조체 선언 => 가독성을 위해
      3. Instruction 추상 클래스를 선언
      4. Instruction 클래스를 상속 받는 Rtype, Itype, Jtype 클래스를 생성
      5. 어셈블리 명령어를 TCHAR 2차원 배열에 저장하는데, 이를 정해진 문자들로 나눈다. (\_tcstok 함수 사용)
      6. 나눈 문자열들을 명령어형식에 따라 구분하여 instruction 클래스의 객체에 저장
      7. 객체의 생성자 호출시, 입력 받은 10진수 변위나 레지스터들을 2진수로 변환해주는 멤버 함수 호출
      8. 화면에 출력 시에는 변환된 2진수들을 각각의 영역에 출력. 어떤 영역인지도 보여줌(opcode 영역인지, rt 영역인지 등)
   2. 내부 구성
      1. 타이틀 바의 X 버튼이나 마지막 화면의 종료 버튼을 누르면 정말 종료하시겠습니까? 메시지가 뜸
      2. 윈도우의 최소, 최대 크기를 정함
      3. 상단에 있는 단계바를 통해 이전 단계에도 접근 가능. 이후 단계에는 접근할 수 없음 (현재 2단계까지 진행했다면, 1단계로는 다시 갈 수 있지만 3단계로는 갈 수 없음)
      4. 현재 어떤 단계에 있는지 단계바에 검은 원으로 표현
      5. 폰트 크기나 형태 설정을 통한 가독성 향상
      6. 모든 화면들을 반응형으로 설정하여 본인이 원하는 해상도에서 진행할 수 있음
3. 결과 및 분석
   1. 초기 화면



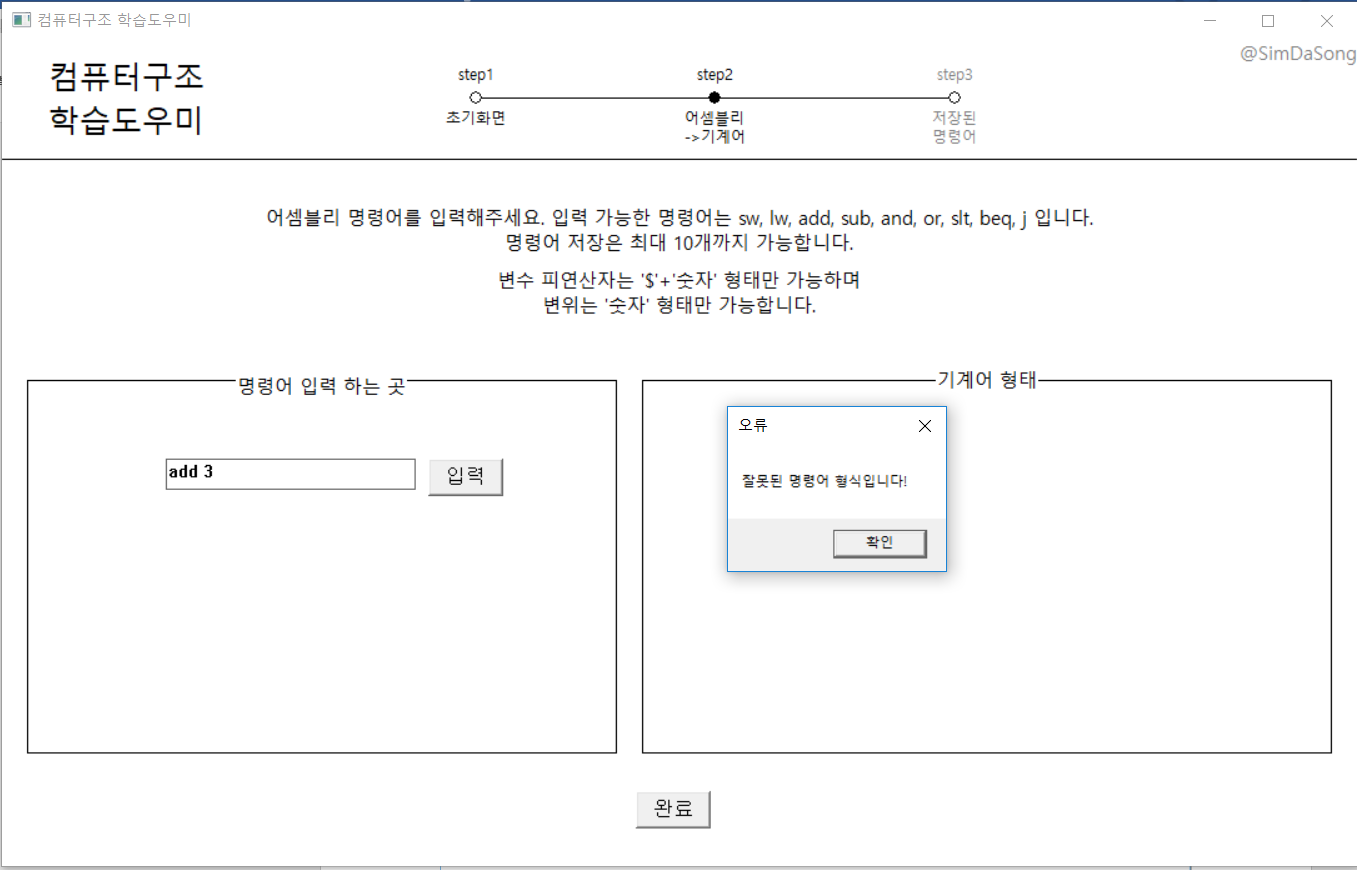
* 1. 시작 버튼을 누른 후



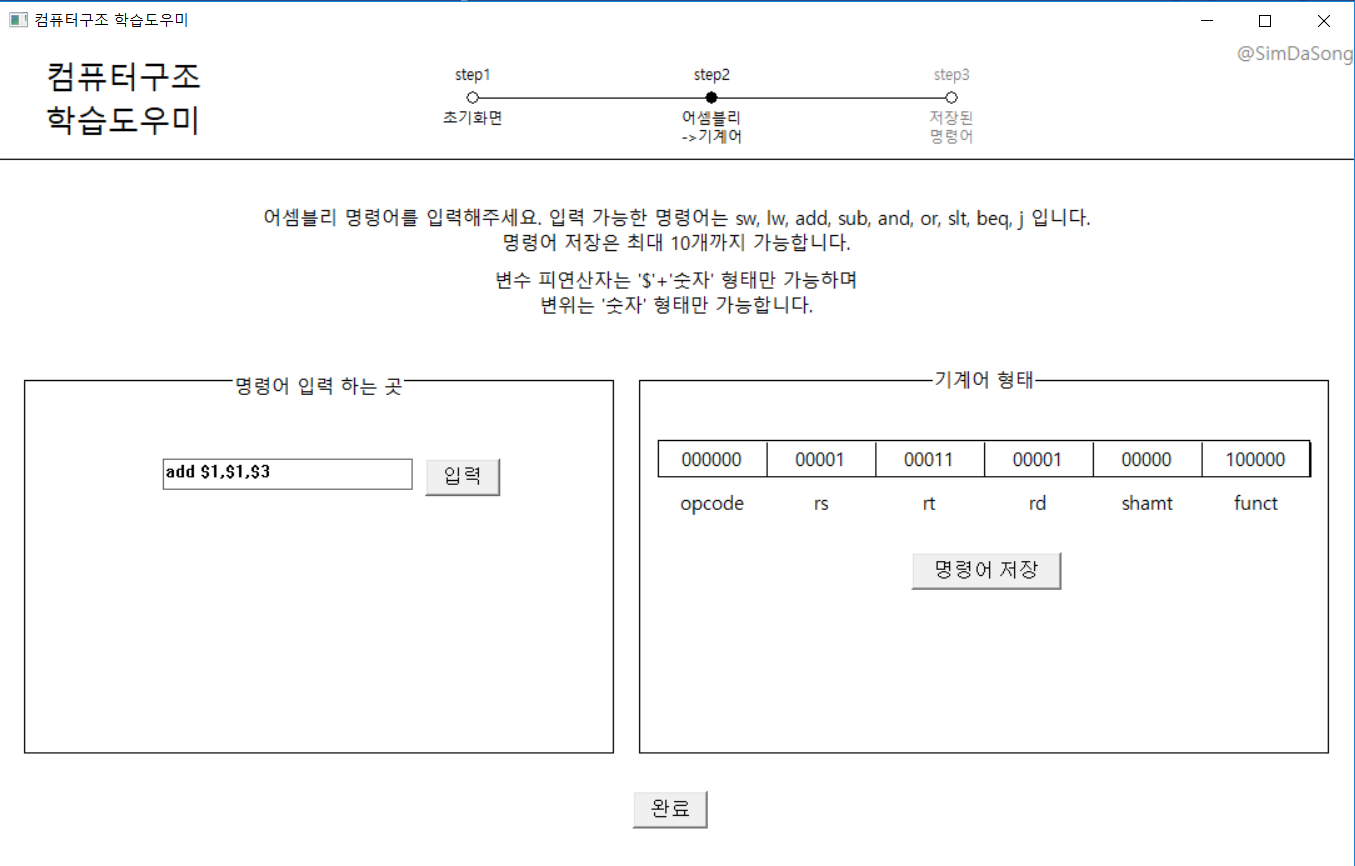
* 1. 상태바의 ‘초기화면’ 텍스트를 클릭하여 1단계로 되돌아 갈 수 있음. ‘어셈블리->기계어’ 텍스트를 클릭하여 2단계로 돌아가는 것도 가능. 하지만 3단계로 가지는 못하게 되어 있음



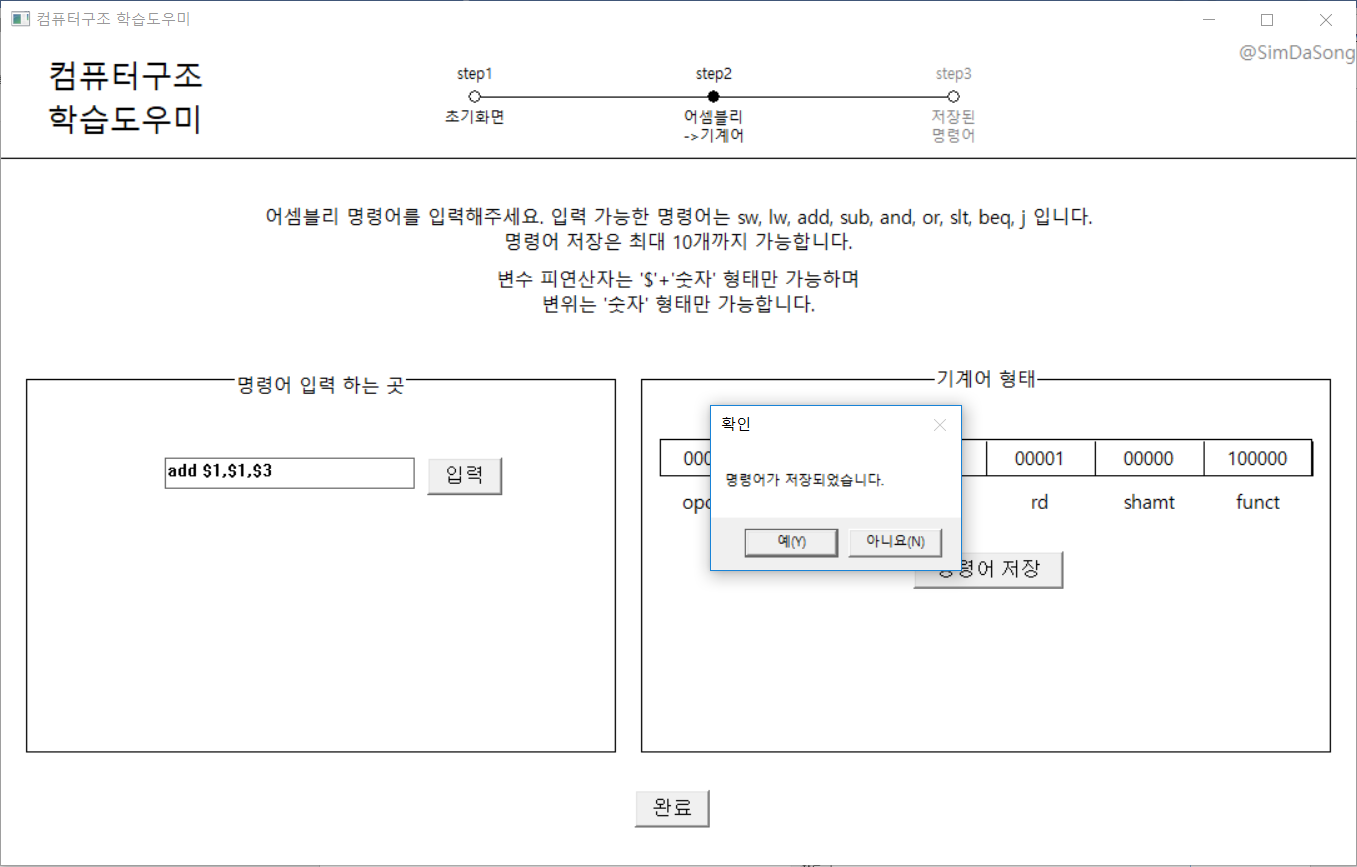
* 1. 주어진 명령어 형식이 아닌 다른 명령어 형식을 입력하면 오류 창이 뜸



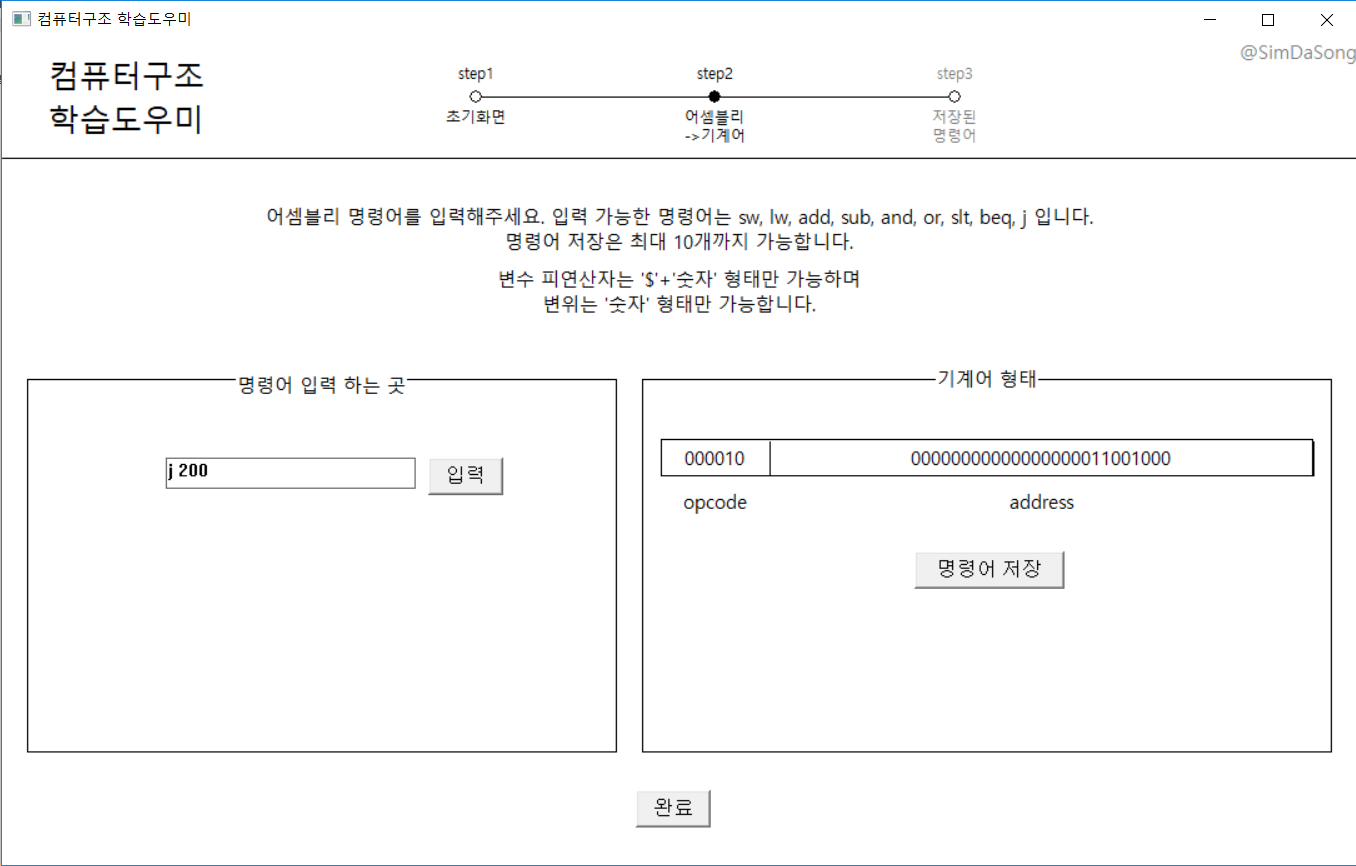
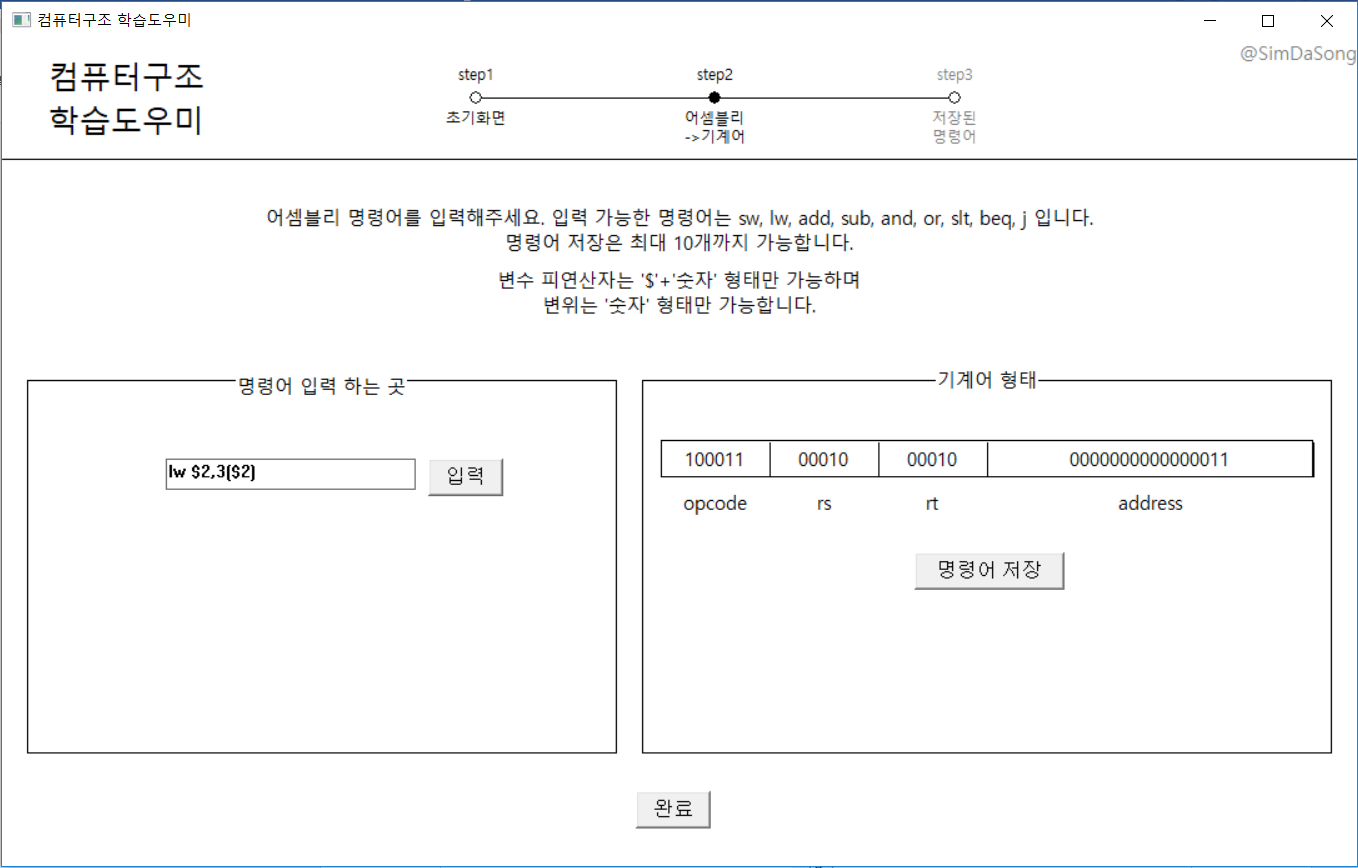
* 1. 제대로 된 명령어를 입력하고 입력 버튼을 누르면 해당 어셈블리언어의 기계어 형태가 각 필드에 나타남. 각 필드가 어떤 필드인지도 알려줌



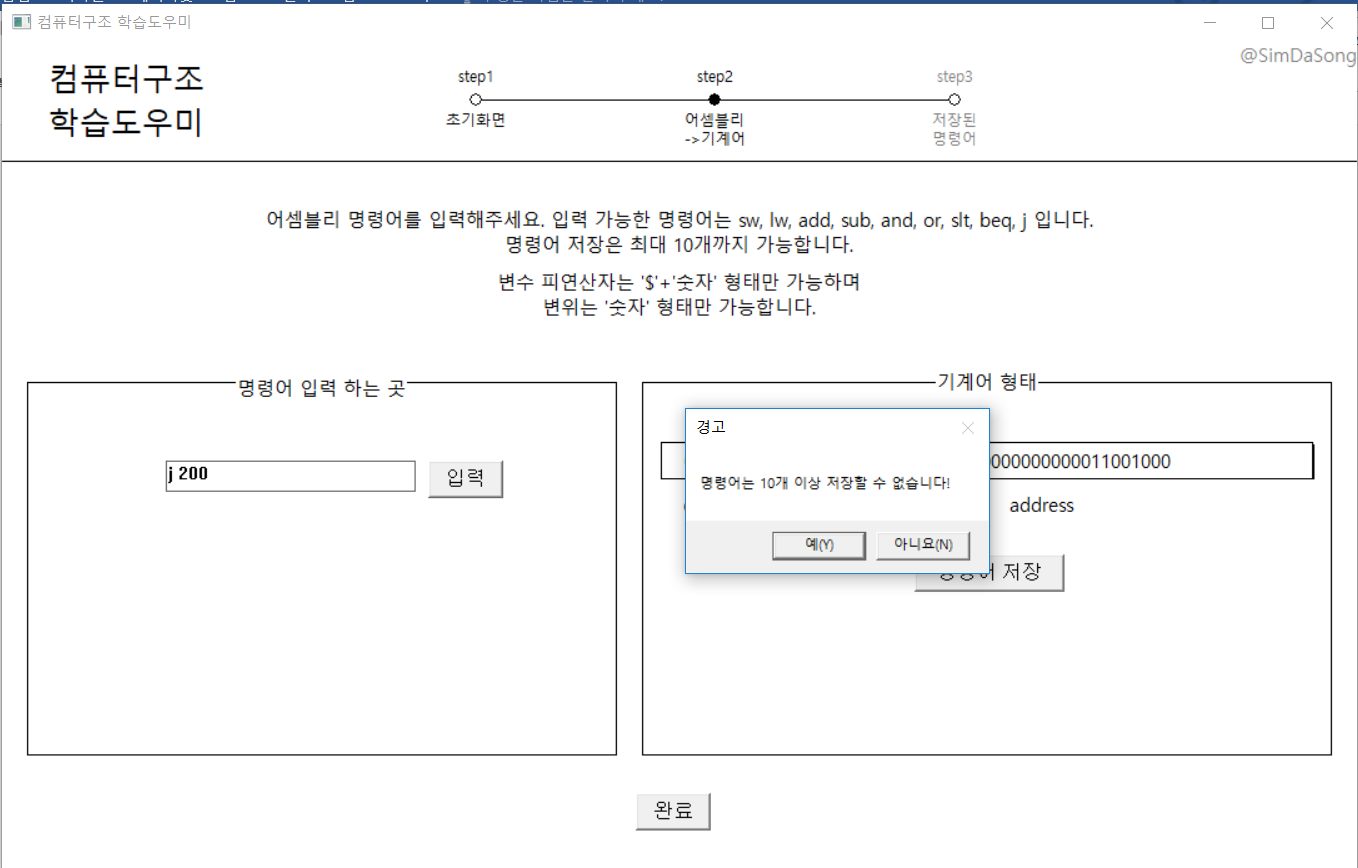
* 1. 명령어 저장 버튼을 누르면 명령어가 저장됨



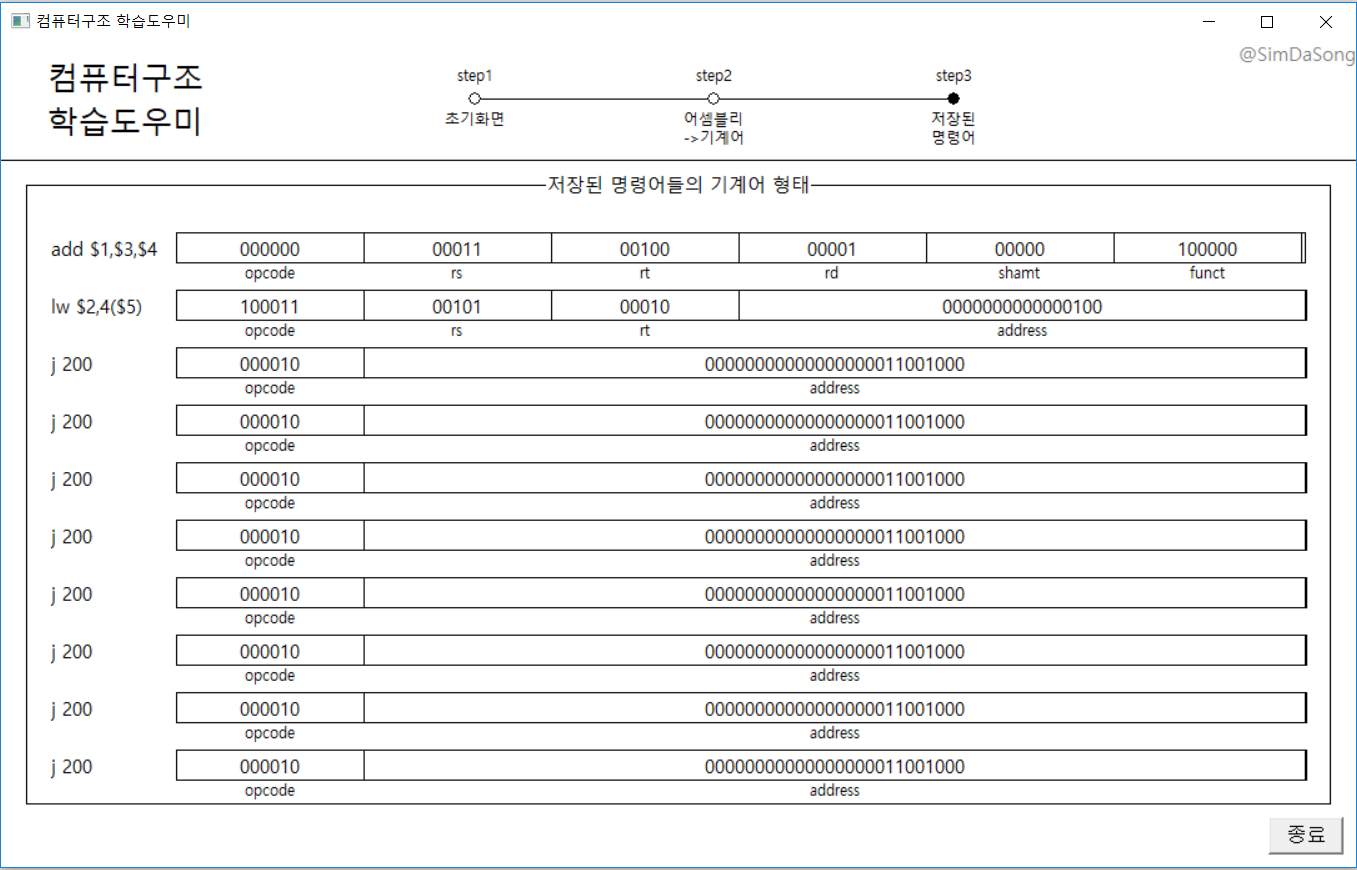
* 1. I 타입 명령어나 J 타입 명령어를 입력하면 필드가 다르게 나옴



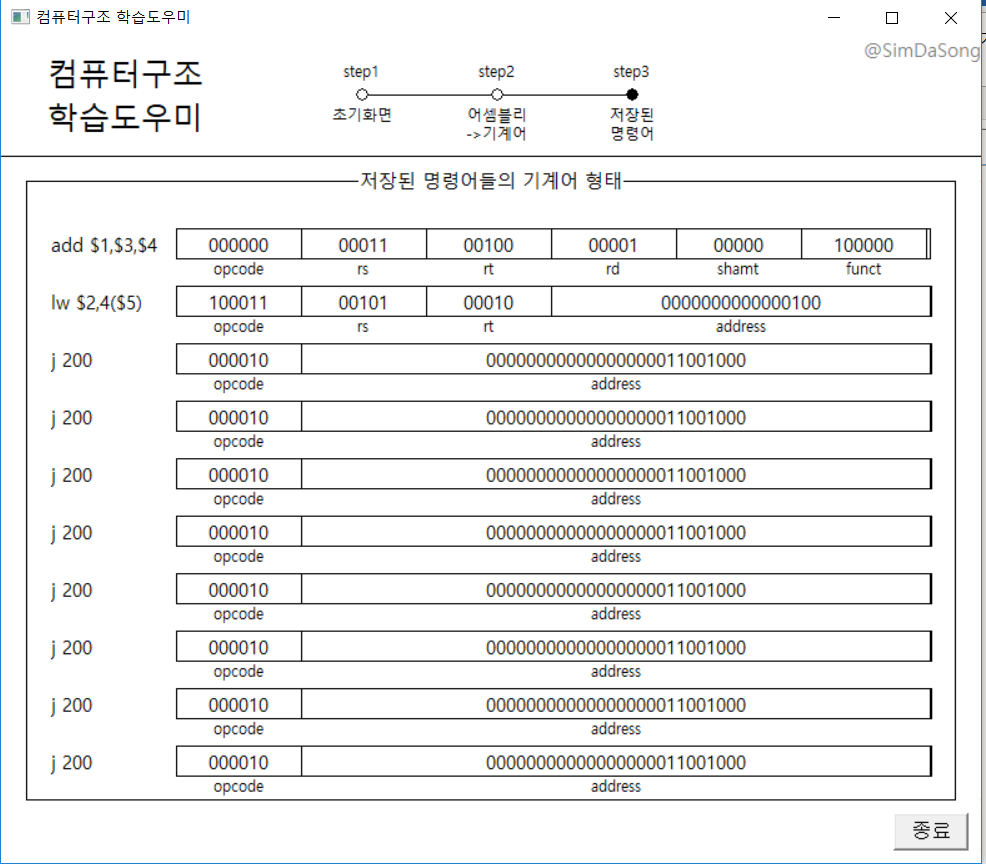
* 1. 명령어는 최대 10개까지 저장 가능하고 10개가 넘으면 오류 창이 뜸



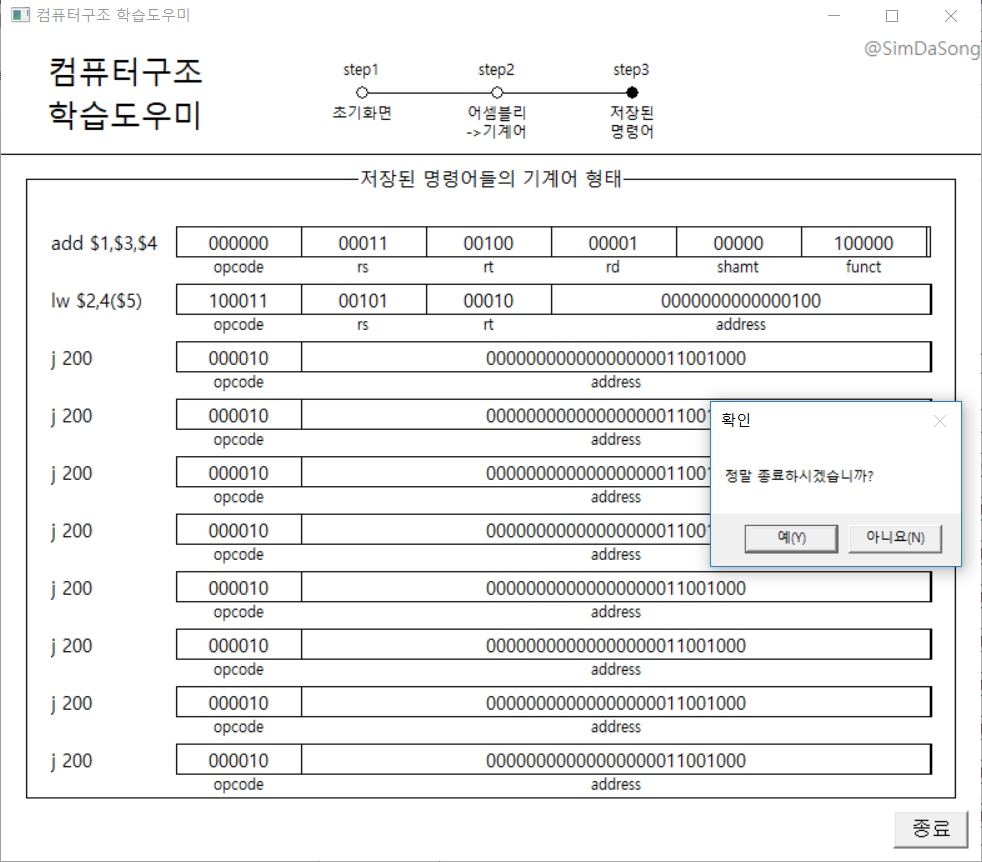
* 1. 완료 버튼을 누르면 3단계로 넘어가며 저장한 명령어의 어셈블리언어 형태와 기계어 형태를 보여줌



* 1. 반응형 화면이라, 화면을 줄이면 그에 맞게 모든 화면이 재조정되고, 최대 화면 크기와 최소 화면 크기가 정해져 있음



* 1. 종료 버튼을 누르면 확인 메시지가 뜨고 예 버튼을 누르면 종료, 아니오 버튼을 누르면 종료 하지 않음



1. 문제점 분석
   1. 완성도 : 초기 프로그램은 파이프라이닝 구현까지 하려고 하였으나 GUI에 집중한 나머지 시간 부족으로 완성하지 못함 => 하지만 어셈블리 명령어 형태를 기계어 형태로 보여주는 것만으로도 초기 목표 (컴퓨터 구조 학습 도움) 성과는 달성했다고 생각
   2. 한정된 수의 명령어의 구현만 가능
2. 향후 보완 필요한 부분
   1. 파이프라이닝까지의 구현
   2. 더 많은 수의 명령어 구현