4.2 버그와 디버깅

개요

버그(bug)는 **코드에 들어있는 오류**입니다. 버그로 인해 프로그램의 실행에 실패하거나 프로그래머가 원하는 대로 동작하지 않게 됩니다. 버그를 만들고 싶지 않겠지만 모든 프로그래머들은 버그와 마주하게 되어있습니다. **디버깅(debugging)**은 **코드에 있는 버그를 식별하고 고치는 과정**입니다. 프로그래머는 **디버거**라고 불리는 프로그램을 사용하여 디버깅을 하게 됩니다.

핵심개념

* 버그
* 디버깅
* 중지점
* GDB

디버깅의 기본

프로그램은 일반적으로 인간보다 훨씬 빠르게 연산을 수행합니다. 그래서 프로그램을 실행시켜보는 것만으로는 무엇이 잘못됐는지 찾아내기 어렵습니다. 디버거는 프로그램을 특정 행에서 멈출 수 있게 해주기 때문에 버그를 찾는데 도움이 됩니다. 프로그래머는 멈춰진 그 지점에서 무슨 일이 일어나는지 볼 수 있습니다. **프로그램이 멈추는 특정 지점**을 **중지점**이라고 합니다. 또한 프로그래머가 프로그램을 한번에 한 행씩 실행할 수 있게 해줍니다. 이로써 프로그래머는 프로그램이 내리는 모든 결정들을 단계별로 따라갈 수 있게 됩니다.

GDB 사용해보기

GDB는 자주 쓰이는 디버거 중 하나입니다. C 프로그램에 GDB을 실행시키려면, 먼저 프로그램을 컴파일해야 합니다. 그런 다음, 보통 때처럼 "./프로그램_이름" 을 치지 말고, **"gdb 프로그램_이름"**을 칩니다.

GDB가 열리면, 가장 먼저 해야할 일은 **중지점을 설정**하는 것입니다. 어디에서 프로그램이 잘못되는지 짐작이 간다면, 그 지점 이전에 있는 행에 중지점을 설정하는 것이 좋습니다. 여러분의 프로그램이 문제가 생길 것이라 생각한 그 지점에 들어서면 어떤 일이 생기는지 볼 수 있기 때문입니다. 어디가 문제인지 확실하지 않다면, 처음부터 모든 코드를 살펴볼 수 있도록 여러분의 main 함수의 첫 행에 중지점을 설정해도 괜찮습니다.

Command	Result
b	add breakpoint
r	runs program
info b	shows breakpoints
clear [#]	removes breakpoint
n	step forward one block
s	step forward one line
p [var]	print variable value
info locals	print local variable values
bt	show function calls
q	quit GDB
c	continue program

▲ 〈GDB 명령어〉

중지점을 설정하기 위해서는 프로그램을 멈추고 싶은 행 번호 다음에 **'b'**를 치고(breakpoint를 의미) 엔터 키를 누릅니다. 이렇게 하면 여러분 프로그램에 중지점이 설정될 겁니다. 현재의 모든 중지점을 보고 싶다면 **"info b"**를 치면 모든 중지점의 위치를 보여줄 것입니다. 중지점의 행 번호 다음에 **"clear"**를 치면 중지점을 제거할 수 있습니다.

중지점을 설정했다면, 'r' (run의 의미)로 프로그램을 실행합니다. 여러분들의 프로그램이 명령어 인자를 받는다면, 'r' 다음에 인자들을 쓰세요. 프로그램이 실행될 것이고, 중지점에서 자동으로 멈출 겁니다. 중지점마다 프롬프트가 나타날 것입니다. 이 때 몇 가지 옵션들이 있습니다.

현재 지점에서 **프로그램의 변수값을 보고 싶다면** 변수 이름 다음에 **'p'**를 입력합니다(print). **"info locals"** 명령어는 **현재 모든 지역 변수값**을 보여줄 것입니다.

코드의 다음 행으로 나아가고 싶다면 **'n'**을 입력합니다(next). **'s'** (step)를 쳐도 코드의 다음 행으로 가기는 하지만, 함수 내부로 들어가서 함수 내부의 각 행을 훑을 것입니다.

프로그램을 계속 실행하고 싶다면 **'c'**를 입력합니다(continue). 중지점이 없다면 프로그램은 종료할 것입니다. 중지점이 있다면, GDB는 다음 중지점에서 멈출 겁니다.