시스템프로그래밍기초 실습

GDB

GDB 개요

- GDB의 개요
  - debugger의 목적
    - 특정 프로그램이 수행하는 동안 내부의 상태(변수, 포인터 값 등)을 관찰하거나, 프로그램이 실행 시 에러가 발생했을 때 원인을 규명하기 위한 도구
  - GDB의 기능
    - 다른 프로그램을 수행할 수 있다.
    - 지정된 위치에서 수행을 멈출 수 있다.
    - 프로그램이 정지했을 때, 그 때의 상태를 확인할 수 있다.
    - 발견된 bug를 임시로 수정하여, 계속해서 다음으로 진행할 수 있다.

GNU Debugger

- GDB의 수행
  - GDB의 시작
    - gdb [프로그램]
    - gdb -help 를 수행하면 option에 대한 설명이 나온다.
    - 컴파일 시에 -g 옵션을 사용해야 한다.

```
% gcc -g myfile.c
```

  - GDB의 종료
    - quit 또는 Ctrl-D를 입력한다.

GNU Debugger

- GDB의 명령어
  - 명령어의 형태
    - 입력은 한 라인으로 받는다.
    - 명령어로 시작하며, 각 명령어에 부속된 인자들이 뒤에 나온다.
    - 명령어는 중복되지 않는 범위에서 약어로 쓸 수 있다.
    - 빈 줄(enter만 입력한 경우)은 이전 명령어의 반복을 나타낸다. (일부 명령어 제외)

GNU Debugger

- Help 기능
  - (gdb) 프롬프트에서 help를 입력하면 명령어에 대한 도움말이 나온다.

```
(gdb) help
List of classes of commands:

running -- Running the program
stack -- Examining the stack
data -- Examining data
breakpoints -- Making program stop at certain points
files -- Specifying and examining files
status -- Status inquiries
support -- Support facilities
```

GNU Debugger

- 프로그램 시작: run 명령어
- 프로그램 인자 전달
  - set args [arglist]
    - argument를 설정한다.
    - arglist가 없을 경우에는 기존의 argument를 없애는 기능을 수행한다.
  - show args : 현재 설정되어 있는 argument를 보여준다.
  - run [arglist]
    - 프로그램을 실행한다.
    - argument를 지정한다.

```
% gdb fio
    GDB is free software and you are welcome to distribute
copies of it
     under certain conditions; type "show copying" to see
the conditions.
    There is absolutely no warranty for GDB; type "show
warranty" for details.
    GDB 4.13 (sparc-sun-sunos4.1.2),
    Copyright 1994 Free Software Foundation, Inc…
    (gdb) run [프로그램의 argument]
```

GNU Debugger

- breakpoint 설정: break 명령어
  - break *[file:]function*
  - break *[file:]linenum*
  - break +*offset*
  - break -*offset*
- breakpoint 해제: clear, delete 명령어
  - clear *[file:]function*
  - clear *[file:]linenum*
  - delete *[breakpoint 번호]*
    - breakpoint 번호를 지정하지 않으면 모든 breakpoint가 해제된다.
- info break
  - 현재 설정된 breakpoint에 대한 정보를 보여 준다.

GNU Debugger

- 프로그램의 진행: continue, step, next
  - continue: 다음의 breakpoint 직전 까지 수행
  - step: 소스 프로그램의 한 줄 진행 (함수인 경우 진입)
  - next: 소스 프로그램의 한 줄 진행
  - kill: 프로그램의 수행을 종료한다.
- 데이터의 내용 출력: print, x, display
  - print: 현 상태에서 변수의 값을 출력한다.(ex. print var1)
  - display: 설정된 변수의 값을 매번 출력한다.
    - delete display *arg1* : *arg1*의 display 해제 (delete display 1)
    - info display : 현재 설정되어 있는 display의 정보를 보여 준다.
    - undisplay *display 번호* : 지정된 display를 해제한다.
- 소스 출력: list 명령어

gdb 실행

- gdb [options][objfile[corefile|process-id]]
- Options
  - help (h)
  - x file : execute GDB commands from file
  - directory (d) : add directory for source files
  - quiet(q) : do not print the introductory and coyright messages

GDB 명령어

- 실행 및 트레이스
  - `run` : 현재의 인수를 사용하여 프로그램을 실행
  - `run <args>` : 새로운 <인수>를 가지고 프로그램을 실행
  - `continue` : 현재 위치에서 프로그램을 계속 실행 (약자 `c`)

GDB 명령어

- next : 한 줄씩 실행 시킨다. 이 때 함수를 포함하고 있으면 함수를 수행시킨다. (약자) n
- next <n> : <n>줄을 실행시킨다.
- step : 한 줄씩 실행 시킨다. 이 때 함수를 포함하고 있으면 함수 내부로 들어가서 한 줄씩 실행한다. (약자) s
- step <n> : <n>줄을 실행시킨다.
- **(주의) step 사용시 원치않은 함수로 들어가지 않도록 주의해야함**

GDB 명령어

- break <line number> : 라인 번호에서 프로그램 실행을 멈추게 한다.
- break <함수 명> : 함수 내부의 첫번째 라인에서 프로그램의 실행을 멈추게 한다.
- quit : gdb를 종료 시킨다.

GDB 명령어

- 데이타에 관련된 명령들
  - whatis <expr> : 지정한 <변수>에 관련된 정보를 보여준다.
  - print <expr> : <expr>에 지정된 식의 값을 보여준다. (약자) p
  - display : 현재 지정된 display 명령의 목록을 보여준다.
  - undisplay
  - disable display
  - enable display
  - display <expr> : 새로운 <expr>을 display목록에 추가

GDB 명령어

- list : 현재 위치에서 소스 파일의 내용을 10줄 보여준다.
- list <first>,<last> : <시작줄>과 <끝줄>사이의 소스파일 내용을 보여준다.

- help [name] : 명령어의 사용방법 안내

- quit : gdb의 종료

간단한 정리

명령어	의 미
b (breakpoint)	실행 중 디버그를 위해 멈추는 위치 지정
b 함수명	함수명에서 멈춤
b 라인번호	라인번호에서 멈춤
r (run)	실행 시작
n (next)	현재 라인 실행 (함수의 경우 실행하고 다음 라인으로 넘어 감)
s (step)	현재 라인 실행 (함수의 경우 호출된 함수 내로 들어가 실행 계속)
c (continue)	다음 breakpoint까지 실행
l (list)	현재 수행되고 있는 라인부터 10개 라인씩 연속적으로 소스 코드를 프린트
p (print) 변수명	변수명으로 저장되어 있는 내용을 프린트
h (help)	도움말
q (quit)	gdb 종료

GDB의 사용예

- 예제프로그램

```
1 #include <stdio.h>
2
3 main()
4 {
5       int i;
6       double j;
7       char *bug = NULL;
8
9       for(i = 0; i < 5; i++) {
10          j = i/2 + i;
11          printf(" j is %lf \n", j );
12      }
13      strcpy(bug,"hi");
14      printf("bug is %s \n", bug);
15
16      return;
17 }
```

- 파일 이름 : test.c
- 버그(bug)가 눈에 보이나요?

GDB의 사용예

- 컴파일과 실행

```
]$ cc -g test.c
]$ ./a.out
 j is 0.000000
 j is 1.000000
 j is 3.000000
 j is 4.000000
 j is 6.000000
Segmentation fault (core dumped)
]$
]$ ls
 a.out  core  test.c
```

GDB의 사용예

- GDB의 시작
  - $ gdb <실행화일>

```
]$ gdb a.out
GNU gdb 4.17.0.11 with Linux support
Copyright 1998 Free Software Foundation, Inc.
GDB is free software, covered by the GNU General
Public License, and you are
welcome to change it and/or distribute copies of it
under certain conditions.
Type "show copying" to see the conditions.
There is absolutely no warranty for GDB.  Type "show
warranty" for details.
This GDB was configured as "i386-redhat-linux"...
(gdb)
(gdb)
```

GDB의 사용예

- list명령

```
(gdb) list
1   #include <stdio.h>
2
3   main()
4   {
5       Int i;
6       double j;
7       char *bug = NULL;
8
9       for( i = 0; i < 5; i++) {
10          j = i/2 + i;
(gdb)
```

```
(gdb) list 4, 13
4   {
5       int i;
6       double j;
7       char *bug = NULL;
8
9       for( i = 0; i < 5; i++) {
10          j = i/2 + i;
11          printf(" j is %lf \n", j );
12      }
13      strcpy(bug,"hi");
(gdb)
```

GDB의 사용예

- break, run 명령
  - break point의 지정, run으로 break point까지 실행

```
(gdb) break 9
Breakpoint 1 at 0x804840d: file test.c, line 9.
(gdb) run
Starting program: /home/users/phd/sugar/bit/gdb/a.out

Breakpoint 1, main () at test.c:9
9               for( i = 0; i < 5; i++) {
(gdb)
```

GDB의 사용예

- next, print 명령

```
(gdb) n
10      j = i/2 + i;
(gdb) p i
$1 = 0
(gdb) p j
$2 = 4.8699524093964861e-270
(gdb) n
11      printf(" j is %lf \n", j );
(gdb) p i
$3 = 0
(gdb) p j
$4 = 0
(gdb) n
 j is 0.000000
9       for( i = 0; i < 5; i++) {
(gdb)
```

GDB의 사용예

- display 명령

```
(gdb) display i
1: i = 0
(gdb) display j
2: j = 0
(gdb) n
10        j = i/2 + i;
2: j = 0
1: i = 1
(gdb) n
11        printf(" j is %lf \n", j );
2: j = 1
1: i = 1
```

```
(gdb) n
 j is 1.000000
9       for( i = 0; i < 5; i++) {
2: j = 1
1: i = 1
(gdb) n
10          j = i/2 + i;
2: j = 1
1: i = 2
(gdb) n
11          printf(" j is %lf \n", j );
2: j = 3
1: i = 2
(gdb)
```

GDB의 사용예

- …next 명령어 계속...

```
(gdb) n
10          j = i/2 + i;
2: j = 4
1: i = 4
(gdb) n
11          printf(" j is %lf \n", j );
2: j = 6
1: i = 4
(gdb) n
 j is 6.000000
9           for( i = 0; i < 5; i++) {
2: j = 6
1: i = 4
```

```
(gdb) n
13      strcpy(bug,"hi");
2: j = 6
1: i = 5
(gdb) n

Program received signal SIGSEGV,
Segmentation fault.
strcpy (dest=0x0, src=0x80484ec
"hi")
at ../sysdeps/generic/strcpy.c:38
../sysdeps/generic/strcpy.c:38:
No such file or directory.
(gdb) q
```

GDB의 사용예

```
]$ gdb a.out
GNU gdb 4.17.0.11 with Linux support
…생략...
(gdb) b 13
Breakpoint 1 at 0x8048460: file test.c,
line 13.
(gdb) run
Starting program:
/home/users/phd/sugar/bit/gdb/a.out
 j is 0.000000
 j is 1.000000
 j is 3.000000
 j is 4.000000
 j is 6.000000
Breakpoint 1, main () at test.c:13
13              strcpy(bug,"hi");
(gdb) p bug
$1 = 0x0
(gdb)
```

- bug의 값(주소)가 잘못되어 있다!

GDB의 사용예

- 예제프로그램
  - (vi debug.c에서 set number 명령 실행)

```
_____1_#include <stdio.h>
_____2_
_____3_void
_____4_print_sum(sum)
_____5_int      sum;
_____6_{
_____7_         printf("Total sum : %d\n", sum);
_____8_}
_____9_                 _____10_main()
                        _____11_{
                        _____12_        int     i, sum;
                        _____13_
                        _____14_        sum = 0;
                        _____15_        for(i=0;i<5;i++) {
                        _____16_                printf("%dth interation\n", i );
                        _____17_                sum += i;
                        _____18_        }
                        _____19_        print_sum(sum);
                        _____20_}
```

GDB의 사용예

-]$ gcc -g debug.c
-]$ gdb a.out

```
GNU gdb 4.17.0.11 with Linux support
Copyright 1998 Free Software
Foundation, Inc.
....생략....
(gdb) break 14
Breakpoint 1 at 0x80483ee: file
debug.c, line 14.
(gdb) run
Starting program:
/home/users/phd/sugar/bit/a.out
```

```
Breakpoint 1, main () at
debug.c:14
14sum = 0;
(gdb) next
15for(i=0;i<5;i++) {
(gdb) next
16printf("%dth interation\n", i );
(gdb) n
0th interation
17sum += i;
(gdb)
```

GDB의 사용예

```
(gdb) n
15for(i=0;i<5;i++) {
(gdb) n
16printf("%dth interation\n", i );
(gdb) n
1th interation
17sum += i;
(gdb) n
15for(i=0;i<5;i++) {
(gdb) n
16printf("%dth interation\n", i );
(gdb) n
2th interation
17sum += i;
```

```
(gdb) n
15for(i=0;i<5;i++) {
(gdb) n
16printf("%dth interation\n", i );
(gdb) n
3th interation
17sum += i;
(gdb)
```

GDB의 사용예

```
(gdb) n
15for(i=0;i<5;i++) {
(gdb) print sum
$1 = 6
(gdb) print i
$2 = 3
(gdb) display sum
1: sum = 6
(gdb) display i
2: i = 3
```

```
(gdb) next
16printf("%dth interation\n", i );
2: i = 4
1: sum = 6
(gdb) next
4th interation
17sum += i;
2: i = 4
1: sum = 6
(gdb)
```

GDB의 사용예 cont'd

```
(gdb) next
15for(i=0;i<5;i++) {
2: i = 4
1: sum = 10
(gdb) next
19print_sum(sum);
2: i = 5
1: sum = 10
```

```
(gdb) next
Total sum : 10
20}
2: i = 5
1: sum = 10
(gdb) c
Program exited with code 017.
(gdb) quit
[sugar@hussein bit]$
```

버그가 없이 동작함을 확인!

Exercise 1: Debug

GDB를 이용하여 다음 프로그램을 분석하고, 에러를 찾아 올바르게 수정하여라.
(GDB를 이용하여 분석한 과정을 스크린샷, 혹은 텍스트 형식으로 복사하여 수정한 코드와 함께 제출)

```
#include <stdio.h>
#include <string.h>

int main(void)
{
    char *p1 = "abc", *p2 = "pacific sea";

    printf("%s\t%s\t%s\n", p1, p2, strcat(p1, p2));
    return 0;
}
```