1 A Brief Introduction To GDB

GDB, the GNU Project debugger, allows you to see what is going on 'inside' another program while it executes—or what another program was doing at the moment it crashed.

GDB can do four main kinds of things (plus other things in support of these) to help you catch bugs in the act:

- Start your program, specifying anything that might affect its behavior.
- Make your program stop on specified conditions.
- Examine what has happened, when your program has stopped.
- Change things in your program, so you can experiment with correcting the effects of one bug and go on to learn about another.

The program being debugged can be written in Ada, C, C++, Objective-C, Pascal (and many other languages). Those programs might be executing on the same machine as GDB (native) or on another machine (remote). GDB can run on most popular UNIX and Microsoft Windows variants.

`http://www.gnu.org/software/gdb/`

Compile your source files(s) with the `-g` option:

C++: `g++ -g file.cpp` or

C: `gcc -g file.c`

Start `gdb` with your program:

```
gdb a.out
```

1.1 `gdb` Commands

Command	Description
`quit | q`	quit `gdb`
`run | r`	run program
`help`	help
`where`	current location

`display varName`
`disp varName`
`d varName`
	display value of a variable

`break name` or `b num`	set a *breakpoint* name or line number
`next | n`	execute next statement
`list`	display lines near current location
`step | s`	one step (step into function)

There are many more commands. These will take you a long way in your debugging efforts.

1.2 Common Errors in Programs

There are several common errors that occur during program execution:

- Divide by zero
- Infinite loop
- Array bounds
- Partial functionality
- `NULL` pointer

1.3 Divide by zero

```c
/* divZero.c  */

#include <stdio.h>

int main()
{
    int x, y;

    y = 123;
    for( x = 10 ; x >= 0 ; x-- )
        y = y/x;

    printf( "y:%d\n", y );

    return 0;
}
```

```
% gdb a.out
GNU gdb 6.3.50-20050815 (Apple version gdb-1346) (Fri Sep 18 20:33:58 UTC 2009)
Copyright 2004 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-apple-darwin"...Reading symbols for shared librar

(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Reading symbols for shared libraries +. done

Program received signal EXC_ARITHMETIC, Arithmetic exception.
0x00001f4f in main () at divZero.c:10
10        y = y/x;
```

```
(gdb) disp x
1: x = 0
(gdb) disp y
2: y = 0

(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: CS_121/TestPrograms/Debugging/a.out

Program received signal EXC_ARITHMETIC, Arithmetic exception.
0x00001f4f in main () at divZero.c:10
10       y = y/x;
2: y = 0
1: x = 0
(gdb) list
5 {
6    int x, y;
7
8    y = 123;
9    for( x = 10 ; x >= 0 ; x-- )
10       y = y/x;
11
12    printf( "y:%d\n", y );
13
14    return 0;
```

Set a *breakpoint*, then step through the program:

```
(gdb) break 8
Breakpoint 1 at 0x1f37: file divZero.c, line 8.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: CS_121/TestPrograms/Debugging/a.out

Breakpoint 1, main () at divZero.c:8
8    y = 123;
2: y = -1073743396
1: x = -1881141161
(gdb) n  # next statement
9    for( x = 10 ; x >= 0 ; x-- )
2: y = 123
1: x = -1881141161
(gdb) n
10       y = y/x;
2: y = 123
1: x = 10
(gdb) n
9    for( x = 10 ; x >= 0 ; x-- )
2: y = 12
1: x = 10
(gdb) n
10       y = y/x;
2: y = 12
1: x = 9
(gdb) n
9    for( x = 10 ; x >= 0 ; x-- )
2: y = 1
1: x = 9
(gdb) n
10       y = y/x;
2: y = 1
1: x = 8
```

```
....

(gdb) n   # press n until we crash

....

9    for( x = 10 ; x >= 0 ; x-- )
2: y = 0
1: x = 1
(gdb) n
10       y = y/x;
2: y = 0
1: x = 0
(gdb) n

Program received signal EXC_ARITHMETIC, Arithmetic exception.
0x00001f4f in main () at divZero.c:10
10       y = y/x;
2: y = 0
1: x = 0
```

1.4 Infinite loop

```
1  /* infLoop.c  */
2
3  #include <stdio.h>
4
5  int main()
6  {
7      int x, y = 1024;
8
9      for( x = 0 ; x < 10 ; x++ )
10     {
11         y = y + x;
12         if( y > 10 )
13           x--;
14     }
15
16     printf( "y:%d\n", y );
17
18     return 0;
19 }
```

```
(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Reading symbols for shared libraries +. done
^C
Program received signal SIGINT, Interrupt.
main () at infLoop.c:8
8    for( x = 0 ; x < 10 ; x++ )
(gdb) disp x
1: x = -1
(gdb) disp y
2: y = 1024
(gdb) n
10       y = y + x;
2: y = 1024
1: x = 0
(gdb) n
11       if( y > 10 )
2: y = 1024
1: x = 0
(gdb) n
12         x--;
2: y = 1024
1: x = 0
(gdb) n
8    for( x = 0 ; x < 10 ; x++ )
2: y = 1024
1: x = -1
(gdb) n
10       y = y + x;
2: y = 1024
1: x = 0
(gdb) n
11       if( y > 10 )
2: y = 1024
1: x = 0
(gdb) n
12         x--;
2: y = 1024
```

```
1: x = 0
(gdb) n
8    for( x = 0 ; x < 10 ; x++ )
2: y = 1024
1: x = -1
(gdb) n
10       y = y + x;
2: y = 1024
1: x = 0
```

1.5 Array bounds

```c
/* arrayBound.c */

#include <stdio.h>

int main()
{
    int x, y, z[3];

    y = 4321;
    for( x = 10 ; x >= 1 ; x-- )
        z[y] = y/x;

    printf( "z[0]: %d\n", z[0] );

    return 0;
}
```

```
(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Reading symbols for shared libraries +. done

Program received signal EXC_BAD_ACCESS, Could not access memory.
Reason: KERN_INVALID_ADDRESS at address: 0xc0003d30
0x00001f4d in main () at arrayOverFlow.c:10
10        z[y] = y/x;
(gdb) disp y
1: y = 4321
(gdb) disp x
2: x = 10
(gdb) disp z
3: z = {4096, 0, 0}
```

1.6 Partial functionality

```
1  /* partFunc.c */
2
3  #include <stdio.h>
4
5  int main()
6  {
7      int u;
8
9      printf( "Distance:\n" );
10     printf( "   1: feet\n" );
11     printf( "   2: meters\n" );
12     scanf( "%d", &u );
13
14     if( u = 1 )
15       printf( "feet\n" );
16     else if( u == 2 )
17       printf( "meters\n" );
18
19     return 0;
20 }
```

```
(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Reading symbols for shared libraries +. done
Distance:
   1: feet
   2: meters
1
feet

Program exited normally.
(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Distance:
   1: feet
   2: meters
2
feet
```

```
(gdb) break 8
Breakpoint 1 at 0x1eeb: file partFunc.c, line 8.
(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out

Breakpoint 1, main () at partFunc.c:8
8    printf( "Distance:\n" );
(gdb) n
Distance:
9    printf( "   1: feet\n" );
(gdb) n
   1: feet
10    printf( "   2: meters\n" );
(gdb) n
   2: meters
11    scanf( "%d", &u );
(gdb) n
2
13    if( u = 1 )
(gdb) disp u
1: u = 2
(gdb) n
14      printf( "feet\n" );
1: u = 1
```

1.7 NULL pointer

```
1  /* badPointer.c */
2
3  #include <stdio.h>
4  #include <stdlib.h>    /*  memory prototypes  */
5
6  struct node
7  {
8     int info;
9     struct node *next;
10 };
11
12 typedef struct node *NodePtr;
13
14 NodePtr head = NULL;
15 //NodePtr head;
16
17 void AddToFront( int x );
18 int DeleteFront();
19 void PrintList();
```

```
1  int main()
2  {
3      int n;
4
5      AddToFront( 3 );
6      AddToFront( 2 );
7      AddToFront( 1 );
8
9      PrintList();
10
11     n = DeleteFront();
12     printf( "%d\n", n );
13
14     return 0;
15 }
16
17 void AddToFront( int x )
18 {
19     NodePtr n = (NodePtr)malloc(sizeof(struct node));
20     n->info = x;
21     n->next = head;
22
23     //head = n;
24 }
```

```
1  int DeleteFront()
2  {
3      NodePtr p = head;
4      int n = p->info;
5
6      p->next = NULL;
7
8      free( (void *)p );
9  }
10
11
12 void PrintList()
13 {
14     NodePtr p = head;
15
16     printf( "Current list contents: " );
17     printf( "{ " );
18     while( p )
19     {
20         printf( "%3d,", p->info );
21         p = p->next;
22     }
23     printf( "}\n" );
24 }
```

```
% gdb a.out
....

(gdb) run
Starting program: CS_121/TestPrograms/Debugging/a.out
Reading symbols for shared libraries +. done
Current list contents: {  }

Program received signal EXC_BAD_ACCESS, Could not access memory.
Reason: KERN_PROTECTION_FAILURE at address: 0x00000000
0x00001ea8 in DeleteFront () at badPointer.c:48
48    int n = head->info;
(gdb) disp head
1: head = (NodePtr) 0x0
```