2024.11.07	실습 과제	학번	22312072
과제 8	리눅스 프로그래밍 환경	이름	차민경

- **참고사항**

모든 실습 과제는 각 문항에서 요구하는 문제의 명령어와 그 출력 결과를 동시에 기재하여야 합니다.

예: **오늘 날짜를 출력하는 명령어를 쓰시오.**
답:
date
2024. 11. 07. (목) 15:00:00 KST

1. "Hello, Linux!"를 출력하는 "hello.c"를 작성하고, "gdb"로 디버깅이 가능하도록 컴파일하는 "Makefile"을 작성하시오.

답:

```
CC = gcc
OBJECTS=hello.c
CFLAGS = -g
TARGET=hello

${TARGET}: ${OBJECTS}
	$(CC) $(CFLAGS) -o ${TARGET} ${OBJECTS}
```

```
yu22312072@acslab-146:~/8$ make
gcc -g -o hello hello.c

yu22312072@acslab-146:~/8$ ./hello
Hello World!
```

2. [1]에서 작성한 "hello.c"를 "gdb"로 실행하고, "main" 함수의 내용을 출력하시오.

답:

```
yu22312072@acslab-146:~/8$ gdb hello
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from hello...
(gdb) l main
1       #include <stdio.h>
2       main(){
3               printf("Hello World!\n");
4       }
```

3. 강의 자료 3-3의 7~9 페이지를 참고하여 “copy” 함수의 “gdb” 디버깅을 수행하시오.

답:

```
(gdb) b copy
Breakpoint 1 at 0x122a: file copy.c, line 7.
(gdb) info b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x000000000000122a in copy at copy.c:7
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Merry X-mas

Breakpoint 1, copy (from=0x555555558040 <line> "Merry X-mas", to=0x5555555580c0 <longest> "") at copy.c:7
7               i=0;
(gdb) n
8               while ((to[i] = from[i]) != '\0')
(gdb) n
9                       ++i;
(gdb) n
8               while ((to[i] = from[i]) != '\0')
(gdb) n
9                       ++i;
(gdb) n
8               while ((to[i] = from[i]) != '\0')
(gdb) n
9                       ++i;
(gdb) n
8               while ((to[i] = from[i]) != '\0')
(gdb) p to
$1 = 0x5555555580c0 <longest> "Mer"
(gdb) c
Continuing.
Happy New Year!

Breakpoint 1, copy (from=0x555555558040 <line> "Happy New Year!", to=0x5555555580c0 <longest> "Merry X-mas") at
 copy.c:7
7               i=0;
(gdb) p from
$2 = 0x555555558040 <line> "Happy New Year!"
(gdb) n
8               while ((to[i] = from[i]) != '\0')
(gdb) n
9                       ++i;
(gdb) p to
$3 = 0x5555555580c0 <longest> "Herry X-mas"
(gdb) c
Continuing.
Happy New Year![Inferior 1 (process 3522506) exited normally]
```

4. 다음 “segfault.c” 파일을 컴파일하고, “gdb”를 통해 오류가 나는 부분을 찾으시오.

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

int main(void) {
    char *buf;
    strcpy(buf, "Hello, GDB!");

    printf("%s\n", buf);
    return 0;
}
```

답:

```
yu22312072@acslab-146:~/8$ gdb a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from a.out...
(gdb) list
1       #include <stdio.h>
2       #include <string.h>
3
4       int main(void){
5               char *buf;
6               strcpy(buf, "Hello, GDB!");
7
8               printf("%s\n", buf);
9               return 0;
10      }
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Program received signal SIGSEGV, Segmentation fault.
0x0000555555555163 in main () at segfault.c:6
6               strcpy(buf, "Hello, GDB!");
```

5. [4]의 오류를 해결한 C언어 코드를 작성하고, “gdb”를 통해 실행하시오.

답:

```
yu22312072@acslab-146:~/8$ gdb a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from a.out...
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Hello, GDB!
[Inferior 1 (process 3506867) exited normally]
(gdb) list
1       #include <stdio.h>
2       #include <string.h>
3       #include <stdlib.h>
4
5       int main(void){
6               char *buf = (char *)malloc(12 * sizeof(char));
7               if (buf == NULL)
8                       return 1;
9               strcpy(buf, "Hello, GDB!");
10
```

6. 다음 “numbers.c” 파일을 컴파일하고, “gdb”를 통해 변수 “y”의 값을 출력하시오.

```c
#include <stdio.h>

int main(void) {
    int x = 10;
    int y = 20;

    printf("x: %d, y: %d\n", x, y);
    return 0;
}
```

답:

```
yu22312072@acslab-146:~/8$ gdb a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from a.out...
(gdb) list
1       #include <stdio.h>
2
3       int main(void){
4               int x= 10;
5               int y= 20;
6
7               printf("x : %d, y : %d\n", x, y);
8               return 0;
9       }
(gdb) break 6
Breakpoint 1 at 0x1163: file numbers.c, line 7.
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at numbers.c:7
7               printf("x : %d, y : %d\n", x, y);
(gdb) print y
$1 = 20
```

7. 다음 “loops.c” 파일을 컴파일하고, “gdb”를 통해 반복문이 몇 번 실행되는지 확인하시오.

```c
1   #include <stdio.h>
2
3   int main(void) {
4       int i;
5
6       for (i = 1; i < 100; i++) {
7           if (i % 10 == 0) {
8               break;
9           }
10      }
11
12      return 0;
13  }
```

답 :

10번

```
2
3       int main(void){
4               int i;
5
6               for (i=1; i<100; i++){
7                       if (i%10 == 0){
8                               break;
9                       }
10              }
(gdb) break 6
Breakpoint 1 at 0x1131: file loops.c, line 6.
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at loops.c:6
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
6               for (i=1; i<100; i++){
(gdb) step
7                       if (i%10 == 0){
(gdb) step
8                               break;
(gdb) step
12              return 0;
(gdb) step
13      }
```

8. 다음 “ij.c” 파일을 컴파일하고, gdb를 사용하여 “j” 값이 이상한 부분을 찾으시오.

```c
#include <stdio.h>

int main(void) {
    int i;
    double j;

    for (i = 2; i < 5; i++) {
        j = i / 2 + i;
        printf("j is %lf\n", j);
    }

    return 0;
}
```

답:

j를 계산할때 double형이 없기때문에 자동형변환이 안돼서 j가 double형이지만 int 값만 나온다

j = (double)i/2 + i;로 수정을 하면 강제로 i를 double형으로 변환돼서 j의 값이 우리가 원하는 소수점까지 값이 나올것이다

```
yu22312072@acslab-146:~/8$ gdb ./a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(gdb) list
1       #include <stdio.h>
2
3       int main(void){
4               int i;
5               double j;
6
7               for (i=2; i<5; i++){
8                       j = i/2 + i;
9                       printf("j is %lf\n", j);
10              }
(gdb) break 7
Breakpoint 1 at 0x1155: file ij.c, line 7.
(gdb) run
Starting program: /home/yu22312072/8/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at ij.c:7
7               for (i=2; i<5; i++){
(gdb) next
8                       j = i/2 + i;
(gdb) next
9                       printf("j is %lf\n", j);
(gdb) print j
$1 = 3
(gdb) next
j is 3.000000
7               for (i=2; i<5; i++){
(gdb) next
8                       j = i/2 + i;
(gdb) next
9                       printf("j is %lf\n", j);
(gdb) print j
$2 = 4_
```

9. 다음 두 개의 수학 함수를 포함하는 라이브러리 `ex9`를 생성하고 사용하시오.

- 정수 팩토리얼을 반환하는 함수가 포함된 factorial.c
- 최대공약수를 반환하는 함수가 포함된 gcd.c

답:

```
yu22312072@acslab-146:~/8$ cat ex9-factorial.c
int factorial(int n) {
        if (n <= 1)
                return 1;
        return n * factorial(n - 1);
}
yu22312072@acslab-146:~/8$ cat ex9-gcd.c
int gcd(int a, int b) {
        while (b != 0) {
                int temp = b;
                b = a % b;
                a = temp;
        }
        return a;
}
yu22312072@acslab-146:~/8$ cat ex9.c
#include <stdio.h>
#include "ex9.h"
int main()
{
        int num1 = factorial(5);
        int num2 = gcd(10, 5);

        printf("factorial(5) result :  %d\n", num1);
        printf("gcd(10, 5) result : %d\n", num2);
        return 0;
}
yu22312072@acslab-146:~/8$ cat ex9.h
int factorial(int);
int gcd(int, int);
```

```
yu22312072@acslab-146:~/8$ gcc -c ex9-factorial.c ex9-gcd.c
yu22312072@acslab-146:~/8$ ar rcv libex9.a ex9-factorial.o ex9-gcd.o
r - ex9-factorial.o
r - ex9-gcd.o
yu22312072@acslab-146:~/8$ gcc -L. ex9.c -lex9
yu22312072@acslab-146:~/8$ ./a.out
factorial(5) result :  120
gcd(10, 5) result : 5
```

10. 피보나치 함수가 포함된 `fibo.c`를 작성하고, [9]에서 작성한 `libex9.a`에 추가하여 사용하시오.

답:

```
yu22312072@acslab-146:~/8$ cat ex9-fibo.c
int fibo(int n){
        if (n <= 1)
                return n;
        return fibo(n-1) + fibo(n-2);
}
yu22312072@acslab-146:~/8$ cat ex9.c
#include <stdio.h>
#include "ex9.h"
int main()
{
        int num1 = factorial(5);
        int num2 = gcd(10, 5);
        int num3 = fibo(4);

        printf("factorial(5) result :  %d\n", num1);
        printf("gcd(10, 5) result : %d\n", num2);
        printf("fibo(4) result : %d\n", num3);
        return 0;
}
yu22312072@acslab-146:~/8$ cat ex9.h
int factorial(int);
int gcd(int, int);
int fibo(int n){
```

```
yu22312072@acslab-146:~/8$ gcc -c ex9-fibo.c
yu22312072@acslab-146:~/8$ ar rcs libex9.a ex9-fibo.o
```

```
yu22312072@acslab-146:~/8$ gcc -L. ex9.c -lex9
yu22312072@acslab-146:~/8$ ./a.out
factorial(5) result :  120
gcd(10, 5) result : 5
fibo(4) result : 3
```