

# Homework 3-1 (50 pts)

## 1 Floating-point Exception Handling Practice

In homework 3-1, you are required to write a C program to generate the same result as the “An Example in Handlers I” by using the GNU C Library [1]. To be specific, when we discuss trap handlers in the course, the slide provided a Sun’s example in “An Example in Handlers I”.

Here is a summary of how the code is organized:

- The code enables the trap of the “common” floating-point exception cases, i.e. invalid, division by zero, and overflow.
- After that, the program run underflow and overflow floating-point operation in sequence.

Because the code only enables traps and handles the common case, only the overflow floating-point operation will trigger the trap and activate the handler.

### 1.1 Details of the Program Requirement

**We will compile and run your code in the CSIE workstation. Please make sure your program can be compiled and run on the CSIE workstation.** If you don’t have an account please go to CSIE 217 room to apply for one.

- Your program should be organized in the following way:

```
STUDENT_ID/hw3-1.c
STUDENT_ID/makefile
```

The executable title should be called hw3-1. Then zip your STUDENT\_ID folder to STUDENT\_ID.zip

- We will run your program with the following command:

```
unzip STUDNET_ID.zip
cd STUDNET_ID
make
./hw3-1 > out
```

**Please make sure that submitted file can be tested on CSIE workstation by using above commands!**

- In your program, you should output to **stdout**. In your code please **exactly** use the following output formats:

```
printf("min_normal = %g\n", x);
printf("min_normal / 13.0 = %g\n", x);
printf("max_normal = %g\n", x);
printf("max_normal * max_normal = %g\n", x);
```

where x is a **double** type. Each output is respectively for min\_normal, underflow value, max\_normal and overflow value. Your output order should be like

```
min_normal = ***
min_normal / 13.0 = ***
max_normal = ***
max_normal * max_normal = ***
```

where `***` is the value of `x`.

- Additionally, when your program trap the overflow operation, program should output an error message to `stderr` in the following format.

```
fprintf(stderr , "SIGFP is caught!\n");
```

- Please follow the exact output format or your credit will be affected. That is, your code should go on after the handling.

## 1.2 Useful Resources

- Floating-point Exception Handling: [https://gcc.gnu.org/onlinedocs/gcc-3.3.6/g77/Floating\\_002dpoint-Exceptions.html](https://gcc.gnu.org/onlinedocs/gcc-3.3.6/g77/Floating_002dpoint-Exceptions.html)
- FP Exceptions: [https://www.gnu.org/software/libc/manual/html\\_node/FP-Exceptions.html](https://www.gnu.org/software/libc/manual/html_node/FP-Exceptions.html)
- Signal Handling: [https://www.gnu.org/software/libc/manual/html\\_node/Signal-Handling.html](https://www.gnu.org/software/libc/manual/html_node/Signal-Handling.html)
- Non-Local Exits: [https://www.gnu.org/software/libc/manual/html\\_node/Non\\_002dLocal-Details.html](https://www.gnu.org/software/libc/manual/html_node/Non_002dLocal-Details.html)
- Here is a simple example for makefile:

```
all: hw3-1.c
    gcc -o hw3-1 hw3-1.c -lm
```

## References

- [1] “The gnu c libaray.” [Online]. Available: <https://www.gnu.org/software/libc/manual/>