Homework 6

Problem 1

```
#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
void handler(int sig) {
    static int beeps = 0;
    printf("BEEP\n");
     if (beeps < 5) {
         beeps += 1;
        fork();
        alarm(1); /* next SIGALRM will be delivered in 1s */
    } else {
        printf("BOOM!\n");
        exit(0);
    }
}

int main() {
    signal(SIGALRM, handler); /* install SIGALRM handler */
    alarm(1); /* next SIGALRM will be delivered in 1s */
    /* signal handler returns control here each time */
    while (1);
    exit(0);
}
```

1. How many seconds will this program remain approximately?
2. How many BEEPs and BOOMs will be printed if you run the above program?

Problem 2

```
int counter = 2;
void handler1(int sig) {
    counter = counter + 1;
    printf("%d\n", counter);
    exit(0);
}
int main() {
```

```
    signal(SIGINT, handler1);
    printf("%d\n", counter);
    if ((pid = fork()) == 0) {
        while(1) {};
    }
    kill(pid, SIGINT);

    counter = counter - 1;
    printf("%d\n", counter);
    waitpid(-1, NULL, 0);
    counter = counter + 1;
    printf("%d\n", counter);
    exit(0);
}
```

1. The above program validates some guidelines in section 8.5.5, please point out which ones are validated (even if unnecessary) and rewrite the **handler** according to the guidelines (**HINT**: you can use **`Sio_puts`** as thread safe **`printf`** if needed).
2. Please write down all the possible outputs of the original programs.

Problem 3

```
sigset_t sigs;
void handler(int sig) {
    printf("handler\n");
}

int main(void) {
    signal(SIGINT, handler);
    sigemptyset(&sigs);
    sigaddset(&sigs, SIGINT);
    sigprocmask(SIG_BLOCK, &sigs, 0);
    for (int i=0; i<3;i++){
        printf("%d\n",i);
        kill(getpid(),SIGINT);
    }
    sigprocmask(SIG_UNBLOCK,&sigs,0);
    return 0;
}
```

What is the output of the above program? Please list the output and explain your answer.