課題 No.6 の解答例

1. 三回目の Ctrl-C で終了するプログラム

```
#include <signal.h>
#include <unistd.h>

volatile sig_atomic_t flg = 0;  // ハンドラと main で共通のフラグ

void handler3(int sig) {        // 最後のハンドラ
  flg = 1;                      // フラグを立てる
  write(1, "last\n", 5);
}

void handler2(int sig) {        // 第２のハンドラ
  signal(SIGINT, handler3);     // 最後のハンドラに切替える
  write(1, "2nd\n", 4);
}

void handler1(int sig) {        // 第１のハンドラ
  signal(SIGINT, handler2);     // 第２のハンドラに切替える
  write(1, "1st\n", 4);
}

int main() {
  signal(SIGINT, handler1);     // 最初は第１のハンドラを登録する
  while (flg==0) {              // フラグが変化するまでループする
  }
  return 0;
}

/* 実行例
$ a.out
^C1st
^C2nd
^Clast
$
*/
```

2. sleep システムコールと同じ働きをする mysleep 関数

```
#include <stdio.h>
#include <signal.h>
#include <unistd.h>

// バグ：alarm を使用中に mysleep を使用するとタイマーが解除される
void handler(int n) {
}

unsigned int mysleep(unsigned int seconds) {
  if (seconds<=0) return 0;              // 誤動作防止
  sig_t org = signal(SIGALRM, handler);
  alarm(seconds);                        // SIGALRM を予約
  pause();                               // シグナルを待つ
  signal(SIGALRM, org);                  // ハンドラを復元する
  return alarm(0);                       // 他のシグナルで再開する場合に
}                                        //   残り時間を返すことができる

int main() {
  for (;;) {
    printf("hello\n");
    mysleep(1);
  }
  return 0;
}

/* 実行例
$ a.out
hello                  <---- hello が一秒に一度表示される.
hello
hello
hello
hello
^C                     <---- Ctrl-C でプログラムを終了する.
$
```