課題 No.10 の解答例

1. 次々と，環境変数を変更した上で出力をファイルにリダイレクトして date を実行するプログラムを作成しなさい．

```
#include <stdio.h>                                  // perror のため
#include <stdlib.h>                                 // putenv のため
#include <unistd.h>                                 // fork, execve のため
#include <sys/wait.h>                               // wait のため
#include <fcntl.h>                                  // open のため
char *execpath="/bin/date";                         // date プログラムのパス
char *args[] = { "date", NULL };                    // date プログラムの argv
int main(int argc, char *argv[], char *envp[]) {
  for (int i=1; i<argc; i=i+2) {                    // コマンド行引数の各組について
    int pid = fork();                               // 分身 (子プロセス) をつくる
    if (pid < 0) {                                  // fork に失敗した場合は
      perror(argv[0]);                              //   fork のエラーメッセージ
      return 1;                                     //   エラー終了する
    }
    if (pid != 0) {                                 // 親プロセスなら
      int status;                                   //   子プロセスの終了を待つ
      wait(&status);
    } else {                                        // 子プロセスなら
      if (putenv(argv[i]) < 0) {                    //   環境変数を変更する
        perror(argv[i]);                            //   putenv が失敗した場合
        return 1;                                   //   エラー終了する
      }
      if (argv[i+1]!=NULL) {                        //   ファイル名あれば
        close(1);                                   //     標準出力をクローズし
        int fd = open(argv[i+1],                    //       ファイルをオープンし
              O_WRONLY|O_CREAT|O_TRUNC, 0644);      //         リダイレクト完了
        if (fd < 0) {                               //     open がエラーなら
          perror(argv[i+1]);                        //       メッセージを出力し
          return 1;                                 //         エラー終了する
        } else if (fd != 1) {
          fprintf(stderr, "リダイレクト失敗\n");//       原因不明のエラーなら
          return 1;                                 //         エラー終了する
        }
      }
      execv(execpath, args);                        // date プログラムに変身する
      // ここが実行されるならエラーが発生
      perror(execpath);                             // エラーメッセージを出力して
      return 1;                                     // エラー終了する
    }
  }
  return 0;
}
/* 実行例
$ ./a.out TZ=Cuba c.txt TZ=Europe/Rome r.txt
$ cat c.txt r.txt
Thu Jul 19 21:42:40 CDT 2018
Fri Jul 20 03:42:40 CEST 2018
*/
```

2. system 関数のクローン mysystem を作りなさい.

```
// mysytem.c : system 関数もどきとテストドライバ
#include <stdio.h>
#include <unistd.h>    // fork, execXX
#include <sys/types.h> // wait
#include <sys/wait.h>  // wait
int mysystem(char *command) {                        // 課題の関数 mysystem()
  int pid, stat;
  if (command==NULL) return -1;                      // system の仕様に似せる
  pid=fork();
  if (pid<0) return -1;
  if (pid==0) {                                      // 子プロセスは
    execl("/bin/sh", "sh", "-c", command, NULL); //   /bin/sh へ変身
    return 127;                                      //   失敗したら 127 を返す
  } else {                                           // 親プロセスは
    while(wait(&stat)!=pid)                          //   /bin/sh の終了を待つ
      ;
  }
  return stat;                                       // 親プロセスは/bin/sh の
}                                                    //   終了ステータスを返す
// テストドライバ
int main(int argc, char *argv[]) {
  int r;
  if (argc!=2) {
    fprintf(stderr, "使い方 ␣:␣%s␣ コマンド文字列\n", argv[0]);
    return 1;
  }
  r = mysystem(argv[1]);
  printf("r␣=␣%08x\n", r);
  return 0;
}

/* 実行例
$ ./mysystem ls
a.txt      mysystem   mysystem.c
r = 00000000
$ ./mysystem "ls -l"
total 40
-rw-r--r--  1 sigemura  staff     4 Jul 20 10:50 a.txt
-rwxr-xr-x  1 sigemura  staff  8700 Jul 20 10:50 mysystem
-rw-r--r--  1 sigemura  staff  1964 Jul 20 10:50 mysystem.c
r = 00000000
$ ./mysystem "cat a.txt"
abc
r = 00000000
$ ./mysystem "cat aa.txt"                <---- エラーが発生する場合
cat: aa.txt: No such file or directory   <---- cat が表示したエラーメッセージ
r = 00000100
$ ./mysystem "ccat a.txt"                <---- エラーが発生する場合
sh: ccat: command not found              <---- sh が表示したエラーメッセージ
r = 00007f00
*/
```