システムプログラミングII H25年度 前期期末試験 (2013.07.31 重村 哲至)

IE4 ____番 氏名 _____

1. 別紙1のプログラム (ex1.c,ex2.c,ex3.c,ex4.c,ex5.c,ex6.c) を読み問に答えなさい。
(1) 次の実行結果を完成しなさい。(5点×6問=30点)

```
$ ex1
30
20
```

```
$ ex2
10
10
20
```

```
$ ex3
aaa
20
```

```
$ ex4
aaa bbb
```

```
$ ex5
A=B
B=C
C=D
```

```
$ ex6
C
```

(2) ex6.c の次の箇所にエラー処理を追加しなさい。
(5点×2問=10点)

```
pid = fork();

if (pid<0) {
  perror("fork");
  exit(1);
}
```

```
execve("/usr/bin/printenv",a,e);

perror("printenv");
exit(1);
```

2. 別紙2のプログラム (myshell5.c) を読んで答えなさい。
(1) 以下は myshell5 の実行例です。正しい実行例の解答欄に「◯」を、間違っている実行例の解答欄に「×」を記入しなさい。なお、「×」を記入した場合は実行例中の間違っている部分に、アンダーラインを引くこと。但し、最初の date の実行はどの場合も正しいとし、また、表示されている時刻の前後関係等は問題としない。(5点×5問=25点)

実行例 (a)

```
$ myshell5
Command: date
Mon Jul 29 17:18:18 JST 2013
Command: setenv LC_TIME ja_JP.UTF-8
Command: date
2013年 7月29日 月曜日 17時18分18秒 JST
Command: date
2013年 7月29日 月曜日 17時18分18秒 JST
Command:
```

実行例 (a)	◯

実行例 (b)
```
$ myshell5
Command: date
Mon Jul 29 17:18:18 JST 2013
Command: setenv LC_TIME ja_JP.UTF-8
Command: date
2013年 7月29日 月曜日 17時18分18秒 JST
Command: date
Mon Jul 29 17:18:18 JST 2013
Command:
```

実行例 (b)	×

実行例 (c)
```
$ myshell5
Command: date
Mon Jul 29 17:18:18 JST 2013
Command: setenv LC_TIME ja_JP.UTF-8
Command: LC_TIME=C date
2013年 7月29日 月曜日 17時18分18秒 JST
Command: date
2013年 7月29日 月曜日 17時18分18秒 JST
Command:
```

実行例 (c)	×

実行例 (d)
```
$ myshell5
Command: date
Mon Jul 29 17:18:18 JST 2013
Command: setenv LC_TIME ja_JP.UTF-8
Command: LC_TIME=C date
Mon Jul 29 17:18:18 JST 2013
Command: date
2013年 7月29日 月曜日 17時18分18秒 JST
Command:
```

実行例 (d)	○

実行例 (e)
```
$ myshell5
Command: date
Mon Jul 29 17:18:18 JST 2013
Command: setenv LC_TIME ja_JP.UTF-8
Command: LC_TIME=C date
Mon Jul 29 17:18:18 JST 2013
Command: date
Mon Jul 29 17:18:18 JST 2013
Command:
```

実行例 (e)	×

(2) 次の実行例のように、“-c” オプションを指定することでコマンド行引数で書いたコマンドを実行する機能を myshell5 に追加します。myshell5.c の 100 行部分に追加するプログラムを答えなさい。(追加するプログラムは複数行の可能性もあります。) (10 点)

```
$ myshell5 -c "echo aaa"
aaa
$ myshell5 -c "date"
Mon Jul 29 20:17:18 JST 2013
$
```

```
parse(argv[2],args);
if (args[0]!=NULL) execute(args);
```

(3) /bin/sh が持つ (2) で追加した機能とほぼ同等な機能を利用している C 言語関数の名前を答えなさい。(5 点)

C 言語関数の名前	system 関数

(4) 次の実行例になるような myenv プログラムを myshell5.c を参考に完成しなさい。(10 点)

```
$ date
Mon Jul 29 18:06:30 JST 2013
$ myenv LC_TIME=ja_JP.UTF-8 TZ=GMT date
2013年 7月29日 月曜日 09時06分46秒 GMT
$
```

```
/* myenv.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

main(int argc, char *argv[]) {


  int i;
  for (i=1; argv[i]!=NULL; i++)
    if (putenv(argv[i])!=0) break;
  if (argv[i]!=NULL) {
    execvp(argv[i], &argv[i]);
    perror(argv[i]);
  }
  exit(1);
}
```

システムプログラミングII H25年度 前期期末試験 (2013.07.31 重村 哲至)

IE4 ____番 氏名 _____

3. 次の実行例になるような mychout プログラムを完成しなさい。mychout プログラムは標準出力をリダイレクトした上で引数のコマンドを実行する。(注意１：作成するファイルのモードは “rw-r–r–” とする。注意２：エラーが発生した場合は、適切なエラーメッセージを表示すること。) (10 点)

```
$ mychout a.txt date
$ cat a.txt
Mon Jul 29 19:53:30 JST 2013
$
```

```
/* mychout.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <fcntl.h>

main(int argc, char *argv[]) {
  if (argc<3) {
    fprintf(stderr, "Usage : %s file command\n", argv[0]);
    exit(1);
  }


   close(1);
  if (open(argv[1],O_WRONLY|O_TRUNC|O_CREAT,0644)!=1) {
    perror(argv[1]);
    exit(1);
  }
  execvp(argv[2], &argv[2]);
  perror(argv[2]);
  exit(1);
}
```

```
-------------------------------------------------------------
/* ex1.c */
#include <stdio.h>
#include <stdlib.h>

main() {
  int pid, stat, n = 10;
  pid = fork();

  if (pid!=0) {
    n = 20;
    while(wait(&stat)!=pid)
      ;
  } else {
    n = 30;
  }
  printf("%d\n", n);
  exit(0);
}

-------------------------------------------------------------
/* ex2.c */
#include <stdio.h>
#include <stdlib.h>

main() {
  int pid, stat, n = 10;
  pid = fork();

  if (pid!=0) {
    printf("%d\n", n);
    n = 20;
    while(wait(&stat)!=pid)
      ;
    printf("%d\n", n);
  } else {
    printf("%d\n", n);
    n = 30;
    exit(0);
    printf("%d\n", n);
  }
  exit(0);
}

-------------------------------------------------------------
/* ex3.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

main() {
  int pid, stat, n = 10;
  pid = fork();

  if (pid!=0) {
    n = 20;
    while(wait(&stat)!=pid)
      ;
    printf("%d\n", n);
  } else {
    n = 30;
    execlp("echo","echo","aaa",NULL);
    printf("%d\n", n);
  }
  exit(0);
}
```

```
-------------------------------------------------------------
/* ex4.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

char *a[] = {"echo", "aaa", "bbb", NULL};
char **environ;

main() {
  int pid, stat;
  pid = fork();

  if (pid!=0) {
    while(wait(&stat)!=pid)
      ;
  } else {
    execve("/bin/echo", a, environ);
  }
  exit(0);
}

-------------------------------------------------------------
/* ex5.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

char *a[] = {"printenv", NULL};
char *e[] = {"A=B", "B=C", "C=D", NULL};

main() {
  int pid, stat;
  pid = fork();

  if (pid!=0) {
    while(wait(&stat)!=pid)
      ;
  } else {
    execve("/usr/bin/printenv", a, e);
  }
  exit(0);
}

-------------------------------------------------------------
/* ex6.c */
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

char *a[] = {"printenv", "B", NULL};
char *e[] = {"A=B", "B=C", "C=D", NULL};

main() {
  int pid, stat;
  pid = fork();

  if (pid!=0) {
    while(wait(&stat)!=pid)
      ;
  } else {
    execve("/usr/bin/printenv", a, e);
  }
  exit(0);
}
```

```
01: /*
02:  * myshell5.c : IE4 システムプログラミング
03:  *  2013.7.31 平成２５年度前期末試験付録
04:  *  [機能追加版の簡易 shell プログラム]
05:  */
06:
07: #include <stdio.h>
08: #include <stdlib.h>
09: #include <string.h>
10: #include <unistd.h>
11: #include <sys/types.h>
12: #include <sys/wait.h>
13:
14: /*
15:  * parse : 空白区切りのコマンド行を解析し文字列配列に変換
16:  *   buf : コマンド行(入力)
17:  *   args: 文字列配列(出力)
18:  */
19: void parse(char *buf, char *args[]) {
20:   while (*buf!='\0') {
21:     while (*buf==' ' || *buf=='\t')
22:       *buf++ = '\0';
23:     if (*buf=='\0') break;
24:     *args++ = buf;
25:     while (*buf!='\0' && *buf!=' ' && *buf!='\t')
26:       buf++;
27:   }
28:   *args = NULL;
29: }
30:
31: /*
32:  * execute : コマンドを実行する
33:  *    args : コマンド行を構成する文字列の配列
34:  */
35: void execute(char *args[]) {            /* コマンドを実行する       */
36:   /* 内部コマンド */
37:   if (strcmp(args[0],"cd")==0) {        /* cd コマンドなら          */
38:     if (args[1]==NULL)                  /*   引数があるか調べて     */
39:       fprintf(stderr,"引数が不足\n");
40:     else if (chdir(args[1])<0)          /*   親プロセスが chdir する*/
41:       perror(args[1]);
42:   } else if (strcmp(args[0],"setenv")==0) {  /* setenv コマンドなら */
43:     if (args[1]==NULL||args[2]==NULL)   /*   引数があるか調べて     */
44:       fprintf(stderr,"引数が不足\n");
45:     else if (setenv(args[1],args[2],1)<0)
46:       perror("setenv");
47:   } else { /* 外部コマンド */
48:     int pid, status;
49:     if ((pid = fork()) < 0) {           /* 新しいプロセスを作る     */
50:       perror("fork");
51:       exit(1);
52:     }
53:     if (pid==0) {                       /* 子供プロセスなら         */
54:       int i;
55:       for (i=0; args[i]!=NULL; i++)     /* NAME=VAL 形式のものを    */
56:         if (putenv(args[i])!=0) break;  /*   putenv する            */
57:       if (args[i]!=NULL) {
58:         execvp(args[i], &args[i]);      /* コマンドを実行           */
59:         perror(args[i]);                /* 出題時バグ perror(*args) */
60:       }
61:       exit(1);
62:     }
63:     while (wait(&status) != pid)        /* 親は子供の終了を待つ     */
64:       ;
65:   }
66: }
```

```
67:
68: /*
69:  * shell : コマンド行入力、コマンド行解析、コマンド実行を繰り返す
70:  */
71: void shell() {
72:   char buf[1024];                        /* コマンド行は最大1022文字 */
73:   char *args[64];                        /* コマンド行は最大63文字列 */
74:   char *p;
75:   for (;;) {
76:     printf("Command: ");
77:     if (fgets(buf,1024,stdin)==NULL) {  /* コマンド行を入力する */
78:       printf("\n");                      /* EOFなら改行して終了  */
79:       exit(0);
80:     }
81:     if ((p=index(buf, '\n'))!=NULL)     /* 行末の'\n'を取り除く */
82:       *p = '\0';
83:     parse(buf,args);                     /* コマンド行を解析する */
84:     if (args[0]!=NULL) execute(args);   /* コマンドを実行する   */
85:   }
86: }
87:
88: /*
89:  * main : myshell5 はここから実行を開始する
90:  */
91: main(int argc, char *argv[]) {
92:   if (argc==1) {                         /* 引数がないときは          */
93:     shell();                             /*  標準入力を使用して動作  */
94:   } else if (argc>1) {                   /* 引数がある場合は          */
95:     char *args[64];                      /*  argv[2]のコマンドを実行 */
96:     if (argc!=3 || strcmp(argv[1],"-c")!=0) {
97:       fprintf(stderr, "Usage : %s [-c \"command\"]\n", argv[0]);
98:       exit(1);
99:     }
100:     parse(argv[2],args);                /* コマンドを解析し          */
101:     if (args[0]!=NULL) execute(args);   /* コマンドを実行する        */
102:   }
103:   exit(0);
104: }
```