VauLSMorg

06 PT - Processes & Threads

Aneka Soal Ujian Sistem Operasi

Rahmat M. Samik-Ibrahim et.al.

1. **2016-1**

```
/* FORK
 * (c) 2015-2016 M. Anwar Ma'sum and Rahmat M. Samik-Ibrahim
 * This is a free software ------ Rev. 06 - 01-Apr-2016
 */

#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>

void main() {
   pid_t  pid1, pid2, pid3;

   pid1 = pid2 = pid3 = getpid();
   printf(" 2016  2015  2014--\n==================\n");
   printf("[%4d][%4d][%4d]\n", pid1, pid2, pid3);
   fork();
   pid1 = getpid();
   wait(NULL);
   pid2 = getpid();
   if(!fork()) {
      pid2 = getpid();
      fork();
   }
   pid3 = getpid();
   wait(NULL);
   printf("[%4d][%4d][%4d]\n", pid1, pid2, pid3);
}
```

(a) (KOLOM) Lingkari tahun angkatan anda berikut ini: (A) 2016 (B) 2015 (C) lainnya.

(b) (BARIS) Lingkari sesuai angka terakhir (paling kanan) dari NPM anda: 0 1 2 3 4 5 6

(c) Harap mengisi (KOLOM:BARIS) dengan 1000

(d) Harap mengisi kolom dan baris lainnya sesuai dengan keluaran program di atas!

NPM	2016	2015	Lainnya
0	[ ]	[ ]	[ ]
1	[ ]	[ ]	[ ]
2	[ ]	[ ]	[ ]
3	[ ]	[ ]	[ ]
4	[ ]	[ ]	[ ]
5	[ ]	[ ]	[ ]
6	[ ]	[ ]	[ ]

2. **2016-2**

```
/*
 * (c) 2016 Rahmat M. Samik-Ibrahim -- This is free software
 * REV02 Tue Apr 11 19:09:21 WIB 2017
 * START Sun Dec 04 00:00:00 WIB 2016
 * wait()     =  suspends until its child terminates.
 * fflush()   =  flushes the user-space buffers.
 * getppid()  =  get parent PID
 * ASSUME first pid > 2000; first ppid < 1000
 */

#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <sys/wait.h>
#define  NN 2

void main (void) {
   int id1000=getpid()-1000;
   for (int ii=1; ii<=NN; ii++) {
      fork();
      wait(NULL);
      int rPID = getpid()-id1000; // "relative"
      int rPPID=getppid()-id1000; // "relative"
      if (rPPID < 1) rPPID=999;
      printf("Loop [%d] - rPID[%d] - rPPID[%4.4d]\n", ii, rPID, rPPID);
      fflush(NULL);
   }
}
```

Fill the following blanks (program output):

```
Loop [    ] - rPID[                  ] - rPPID[                    ]
Loop [    ] - rPID[                  ] - rPPID[                    ]
Loop [    ] - rPID[                  ] - rPPID[                    ]
Loop [    ] - rPID[                  ] - rPPID[                    ]
Loop [    ] - rPID[                  ] - rPPID[                    ]
Loop [    ] - rPID[                  ] - rPPID[                    ]
```

3. **2017-1**

Program Code of Processes and Threads

```
001 /*
002  * (c) 2005-2017 Rahmat M. Samik-Ibrahim
003  * This is free software. Feel free to copy and/or
004  * modify and/or distribute it, provided this
005  * notice, and the copyright notice, are preserved.
006  * REV02 Wed May 17 16:52:02 WIB 2017
007  * REV00 Wed May  3 17:07:09 WIB 2017
008  *
009  * fflush(NULL): flushes all open output streams
010  * fork():       creates  a new process by cloning
011  * getpid():     get PID (Process ID)
012  * wait(NULL):   wait until the child is terminated
013  *
014  */
015
016 #include <stdio.h>
017 #include <unistd.h>
018 #include <sys/types.h>
019 #include <sys/wait.h>
020 #include <stdlib.h>
021
022 void main(void) {
023    int firstPID = (int) getpid();
024    int   RelPID;
025
026    fork();
027    wait(NULL);
028    fork();
029    wait(NULL);
030    fork();
031    wait(NULL);
032
033    RelPID=(int)getpid()-firstPID+1000;
034    printf("RelPID: %d\n", RelPID);
035    fflush(NULL);
036 }
```

Program Output (line 34 of every process):

RelPID:

---- ----

---- ----

4. (6 points) **2017-2**

The Program Code

```
001 /*
002  * (c) 2017 Rahmat M. Samik-Ibrahim
003  * http://rahmatm.samik-ibrahim.vlsm.org/
004  * This is free software.
005  * REV02 Mon Dec 11 17:46:01 WIB 2017
006  * START Sun Dec  3 18:00:08 WIB 2017
007  */
008
009 #include <stdio.h>
010 #include <unistd.h>
011 #include <sys/types.h>
012 #include <sys/wait.h>
013
014 #define LOOP   3
015 #define OFFSET 1000

017 void main(void) {
018    int basePID = getpid() - OFFSET;
019
020    for (int ii=0; ii < LOOP; ii++) {
021       if(!fork()) {
022          printf("PID[%d]-PPID[%d]\n",
                    getpid()  - basePID,
                    getppid() - basePID);
023          fflush(NULL);
024       }
025       wait(NULL);
026    }
027 }
```

Program Output (line 22 of every process):

---- ----

---- ----

---- ----