Solution

Problem 1: (12 points)

1 Child: B; Parent: A;

2 No. Because the child for fork line 10 do not block the waitpid in line 15. So the output of line 11 may appear at a random time.

3 Yes. Both the parent and child process exit without calling free, and thus have a memory leakage problem. The grandchild process does not have double free or memory leakage problem.

Problem 2: (12 points)

1 [1] 1S [2] 23

2 [3] 0 [4] 4

3 6(commented); 3(uncommented)

Problem 3: (18 points)

1 [1] OBJECT [2] LOCAL [3] 4 [4] FUNC

[5] GLOBAL [6] 1 [7] NOTYPE [8] 0

[9] UND [10] GLOBAL [11] 4 [12] 3

2 value attribute of id is its offset in .data section. (1′)
Before id, there must be names at offset 0, which occupies 16 bytes. (1′)

3 Dongma 103 (2′ for each)

Problem 4: (18 points)

1 [1] id/.bss [2] R_386_PC32 [3] white [4] names

[5] .rodata [6] R_386_PC32 [7] id [8] R_386_32

2. [1] 34 a0 04 08

[2] 83 ff ff ff

[3] 24 a0 04 08

[4] 18 a0 04 08

[5] 15 84 04 08

Problem 5: (7 points)

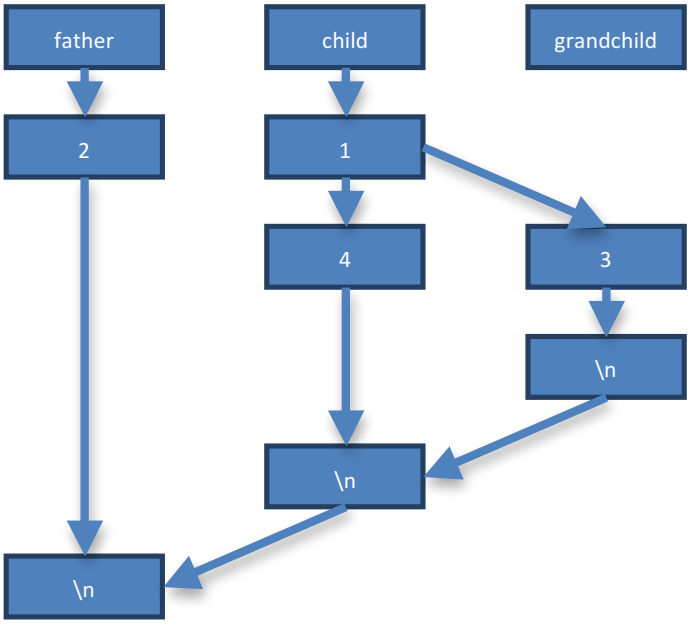
1 .bss section does not occupy space in the ELF files. However, it occupies space in virtual memory.

2 [1] 0x080495a4

[2] 0x0804df08

Problem 6: (15 points)

1.



2.

Count:-1 Count:0 Count:1	Count:-2 Count:0 Count:1	Count:0 Count:1 Count:1
Count:0	Count:-1	

Count:0 Count:1	Count:1 Count:1	

3. kill -7544 SIGUSR2

   kill -7545 SIGUSR2

   kill -7543 SIGUSR1

   or

   kill -7544 SIGUSR2

   kill -7543 SIGUSR1

   kill -7545 SIGUSR2

Problem 7: (18 points)

1

```
int clientfd = Open_clientfd("ipads.se.sjtu.edu.cn", 6666);
sprintf(send_buf, "LABLIST\n");
Rio_writen(clientfd, send_buf, strlen(buf));
```

2

```
char head[20], tmpname[20], taname[20];
rio_t rio;
Rio_readinitb(&rio, clientfd);
int valid = 0;
while (1) {
  Rio_readlineb(&rio, recv_buf, 1024);
  sscanf(recv_buf, "%s %s\n", head, tmpname);
  if (!strcmp(head, "END")) break;
  if (!strcmp(head, labname)) {
    valid = 1;
    strcpy(taname, tmpname);
  }
}
if (!valid){
  printf("Invalid labname!\n");
  Close(clientfd);
  return -1;
 }
```

3

```
    sprintf(send_buf, "INFO %s\n", taname);
    Rio_writen(clientfd, send_buf, strlen(send_buf));
```

4

```
    Rio_readlineb(&rio, recv_buf, 1024);
    printf("%s", recv_buf);
    Close(clientfd);
```