Solution

Problem 1: (11 points)

1 { "/bin/echo", str, 0 }

2 No. Because child processes have their isolated address spaces, each with their own arr variable.

3 3 or 4.

4 processes are created by line 14 in the loop, but waitpid in line 21 only reap 1 of them. The process created by line 12 will not continue to execute line 28~30 after execve, so the other 3 processes created by 14 remains to be zombie in the end.

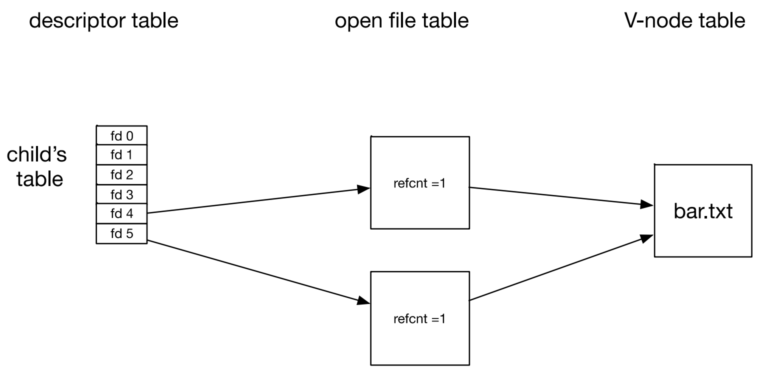
The process created by line 12 might be reaped by waitpid in line 28. However, if it exits after execution of line 28, the while loop in line 28 will break because with the WNOHANG option, waitpid returns 0 if no child process exits. In this case the process created by line 12 will remain zombie at the end.

4 24(possible orders of line 16 outputs) * 4(possible relative positions for line 25 and line 16 outputs) * 6(possible relative positions for line 29 outputs and outputs above) = 576

Problem 2: (13 points)

1 C1

2



3 1212S12 1212CS12 121212CS

4 1C

```
FIX:  printf("%c", c[0]);
      fflush(stdout);
```

Problem 3: (20 points)

1 [1] 26 [2] 3 [3] 3

2

Order	Operation	Hit/Miss
1	0x411488	Miss
2	0x411489	Hit
3	0x411490	Miss
4	0xd844a0	Miss
5	0x178232e	Hit
6	0x13cd48d	Miss

3 9/16

4 4*9/16 + 200*7/16 = 89.75 cycles

5 2-way associate: 50%, 4-way associate: 50%

Problem 4: (23 points)

1. [1] 16 [2] OBJECT [3] GLOBAL [4] COM

[5] 0 [6] NOTYPE [7] UND [8] 0

[9] NOTYPE [10] UND

2. [1] -4 [2] R_X86_64_32 [3] shows

[4] 0 [5] R_X86_64_32 [6] names

[7] 0 [8] R_X86_64_PC32 [9] b

[10] -4

3. [1] 59 00 00 00 [2] 60 10 60 00

[3] d1 0a 20 00 [4] 80 10 60 00 00 00 00 00

4. KUN (3') perform JNTM (2')

Problem 5: (11 points)

1 [1] we can link it when the application is loaded,.

gcc -o prog dynamic_link.c ./libvector.so

[2] we can link when the application executes. Modify the main function.

```
int main() {
        void handle;
        void (*adddvec)(int*, int *, int);
        handle = dlopen("./libvector.so", RTLD_LAZY);
        addvec = dlsym(handle, "addvec");
        subvec(x,y,z,2)
         printf("z = [%d %d]", z[0], z[1]);
        dlclose(handle);
    }
```

2 Because with GOT, the code segment is position-independent(PIC). So a single copy of a shared module's code segment can be shared by an unlimited number of processes.

3 [1] 0x404566. [2] 0x400128

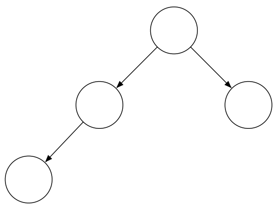
Problem 6: (22 points)

1. No. By default, the kernel blocks pending signals of the type currently being processed by a handler.

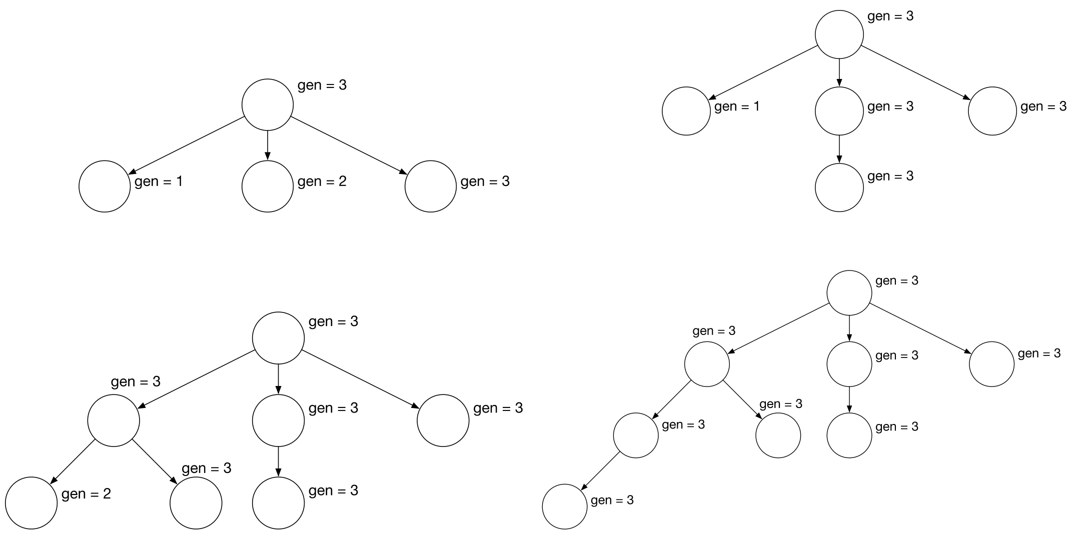
2. No. The memory spaces of the parent and child are isolated.

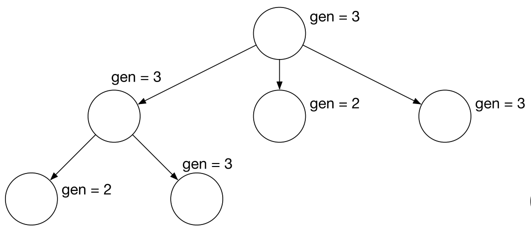
3. a) 2. b) The parent receives twice while the child receives none.

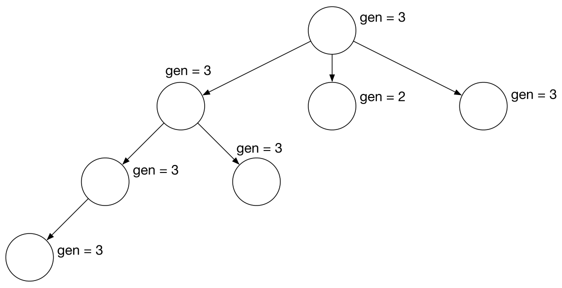
4.



5. Even though every child sets its pgid different with the parent, the signal may be delivered before setpgid. There are 6 cases shown below. The final value of generation in each process is labeled near the circle.




gen = 3
gen = 3
gen = 2
gen = 3
gen = 2
gen = 3


gen = 3
gen = 3
gen = 2
gen = 3
gen = 3
gen = 3
gen = 3