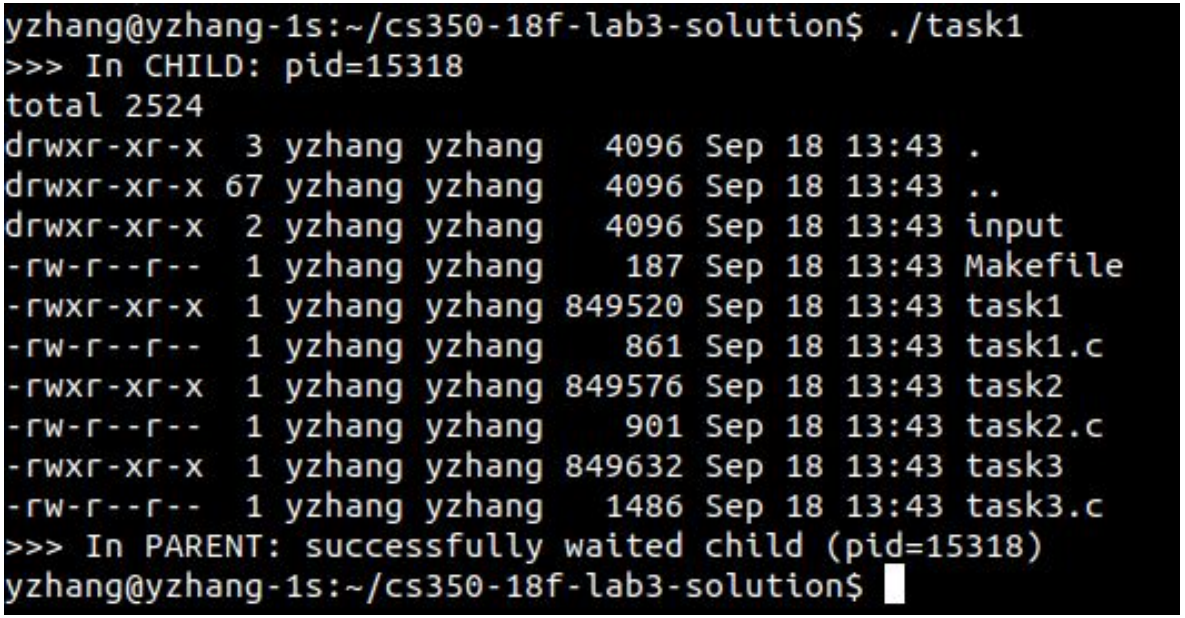
CS 350 Operating Systems
Spring 2023

Lab 4: fork, exec, wait, and dup

Task 1

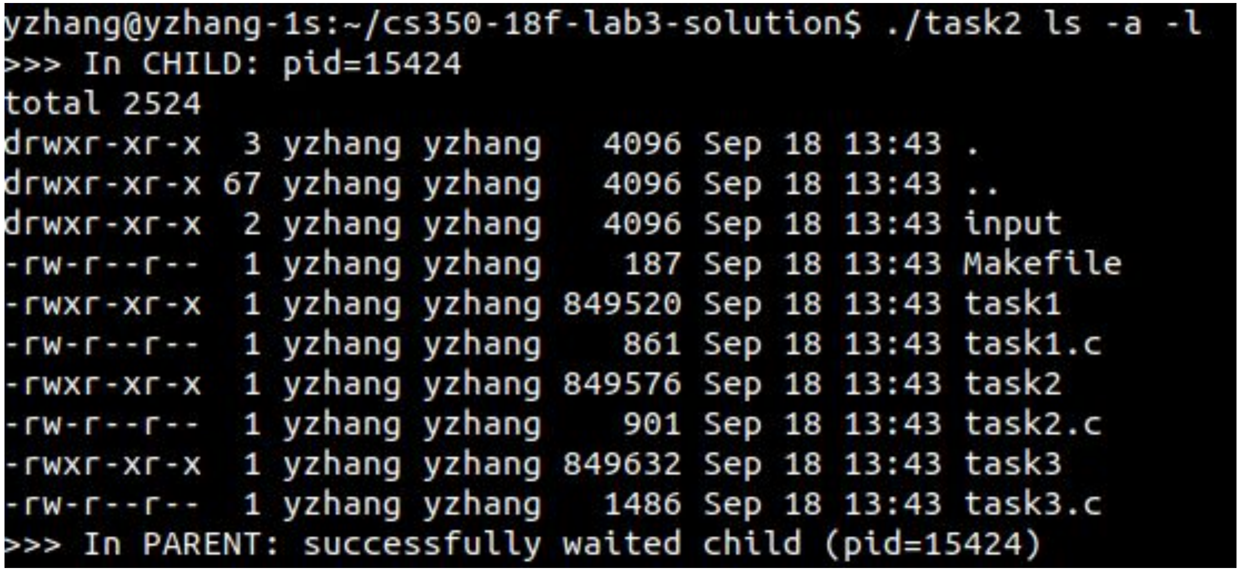
- Write a program which does the following:
  - Main process fork a child process.
  - Child process prints
    "*IN CHILD: pid=child's_actual_pid*"
  - Child process executes the "ls" command with "-l" and "-a" options.
    - This is a hard-coded execution. So using the "l" versions of exec (i.e., `execl()`/`execlp()`) may be more convenient than the "v" versions.
  - Parent waits for the child. Once the child is successfully reaped, parent prints
    "*In PARENT: successfully waited child (pid=child's_pid)*"



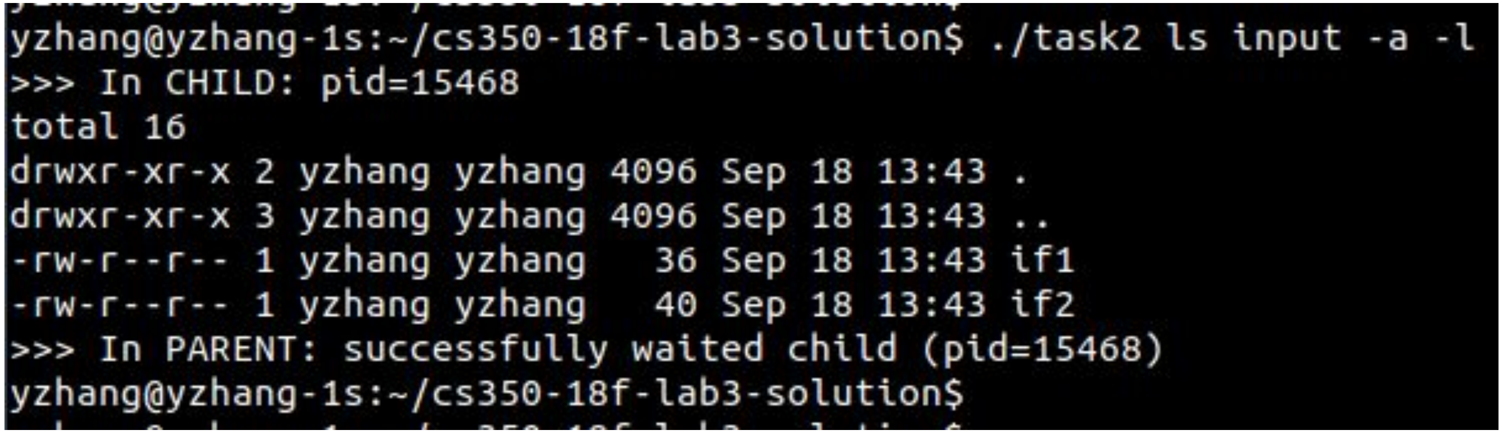
```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task1
>>> In CHILD: pid=15318
total 2524
drwxr-xr-x  3 yzhang yzhang   4096 Sep 18 13:43 .
drwxr-xr-x 67 yzhang yzhang   4096 Sep 18 13:43 ..
drwxr-xr-x  2 yzhang yzhang   4096 Sep 18 13:43 input
-rw-r--r--  1 yzhang yzhang    187 Sep 18 13:43 Makefile
-rwxr-xr-x  1 yzhang yzhang 849520 Sep 18 13:43 task1
-rw-r--r--  1 yzhang yzhang    861 Sep 18 13:43 task1.c
-rwxr-xr-x  1 yzhang yzhang 849576 Sep 18 13:43 task2
-rw-r--r--  1 yzhang yzhang    901 Sep 18 13:43 task2.c
-rwxr-xr-x  1 yzhang yzhang 849632 Sep 18 13:43 task3
-rw-r--r--  1 yzhang yzhang   1486 Sep 18 13:43 task3.c
>>> In PARENT: successfully waited child (pid=15318)
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ 
```

Task 2

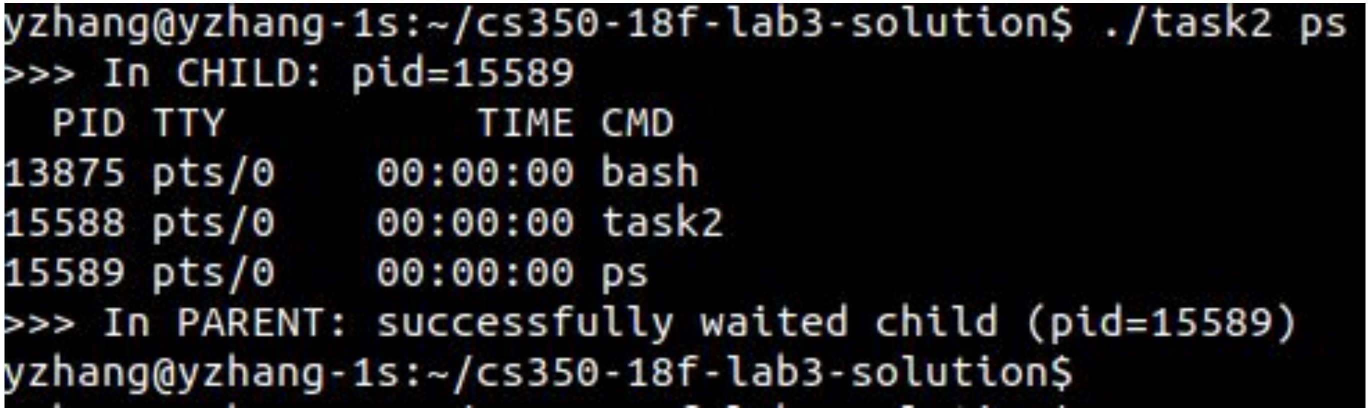
- Similar to task1, but the command to execute and its options are provided by user.
  - See some demos in the next two slides
  - You did an almost the same task previously. So you should know what’s the best practice here.



```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task2 ls -a -l
>>> In CHILD: pid=15424
total 2524
drwxr-xr-x  3 yzhang yzhang   4096 Sep 18 13:43 .
drwxr-xr-x 67 yzhang yzhang   4096 Sep 18 13:43 ..
drwxr-xr-x  2 yzhang yzhang   4096 Sep 18 13:43 input
-rw-r--r--  1 yzhang yzhang    187 Sep 18 13:43 Makefile
-rwxr-xr-x  1 yzhang yzhang 849520 Sep 18 13:43 task1
-rw-r--r--  1 yzhang yzhang    861 Sep 18 13:43 task1.c
-rwxr-xr-x  1 yzhang yzhang 849576 Sep 18 13:43 task2
-rw-r--r--  1 yzhang yzhang    901 Sep 18 13:43 task2.c
-rwxr-xr-x  1 yzhang yzhang 849632 Sep 18 13:43 task3
-rw-r--r--  1 yzhang yzhang   1486 Sep 18 13:43 task3.c
>>> In PARENT: successfully waited child (pid=15424)
```



```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task2 ls input -a -l
>>> In CHILD: pid=15468
total 16
drwxr-xr-x 2 yzhang yzhang 4096 Sep 18 13:43 .
drwxr-xr-x 3 yzhang yzhang 4096 Sep 18 13:43 ..
-rw-r--r-- 1 yzhang yzhang   36 Sep 18 13:43 if1
-rw-r--r-- 1 yzhang yzhang   40 Sep 18 13:43 if2
>>> In PARENT: successfully waited child (pid=15468)
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
```



```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task2 ps
>>> In CHILD: pid=15589
  PID TTY          TIME CMD
13875 pts/0    00:00:00 bash
15588 pts/0    00:00:00 task2
15589 pts/0    00:00:00 ps
>>> In PARENT: successfully waited child (pid=15589)
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
```

Task 3

Similar to task 2, but the difference are

- The input should be taken from the file specified in the "INPUT_FILE" macro in task3.c.
    - Process always tries to read from file descriptor (fd) `STDIN_FILENO` (which is the standard input file descriptor and the value of it is 0) for input.
    - By default, fd `STDIN_FILENO` is associated with terminal input.
    - What you need to do here is to re-associate fd `STDIN_FILENO` with the fd of the actual input file.

Task 3

(cont.)

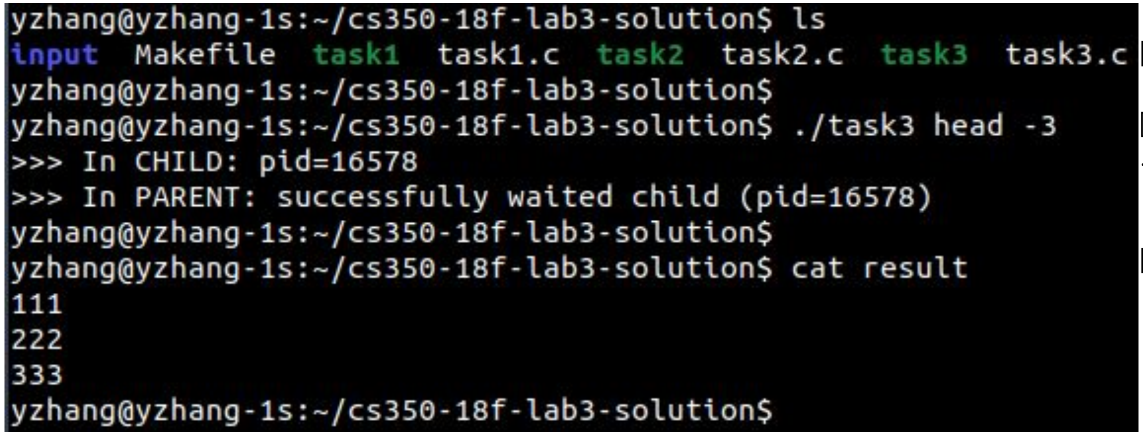
- The output should be written to a file named “result” which locates in the same directory as the “task3” binary.
    - Process always tries to write to file descriptor (fd) `STDOUT_FILENO` (which is the standard output file descriptor and the value of it is 1) for output.
    - By default, fd `STDOUT_FILENO` is associated with terminal output.
    - What you need to do here is to re-associate fd `STDOUT_FILENO` with the fd of the actual output file.

Task 3

(cont.)

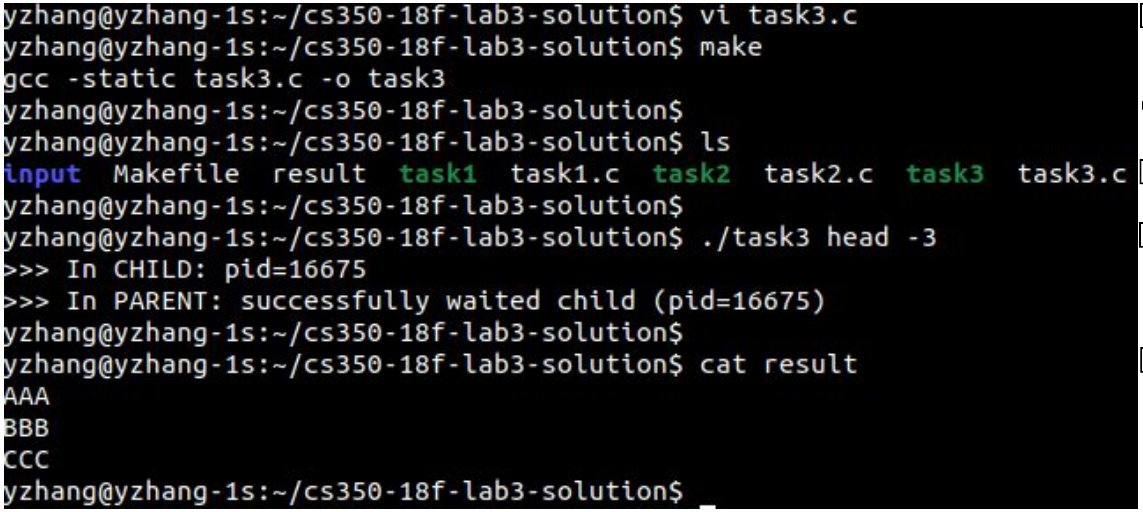
- You will need to use the `dup()`/`dup2()` function for re-associating file descriptors.
  - Read the posted materials for how to use these two functions.
  - Hint: `dup2()` is the better option for this task.

- For how to use `open()` to open a file, read the posted materials also.



```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ls
input   Makefile   task1   task1.c   task2   task2.c   task3   task3.c
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task3 head -3
>>> In CHILD: pid=16578
>>> In PARENT: successfully waited child (pid=16578)
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ cat result
111
222
333
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
```

- No “result”
- Print the first 3 lines of the input, which should be now redirected to “./input/if1”
- Let’s check the content of the file “result”



```
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ vi task3.c
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ make
gcc -static task3.c -o task3
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ls
input   Makefile   result   task1   task1.c   task2   task2.c   task3   task3.c
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ ./task3 head -3
>>> In CHILD: pid=16675
>>> In PARENT: successfully waited child (pid=16675)
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ cat result
AAA
BBB
CCC
yzhang@yzhang-1s:~/cs350-18f-lab3-solution$ _
```

- Change the INPUT_FILE macro to “./input/if2”, and compile.
- Old “result” is there.
- Run it again.
- “result” has been updated based on the new input.