IC221 System Programming
Spring 2015
**HW8**

NAME:_____

COLLABORATOR(S):_____

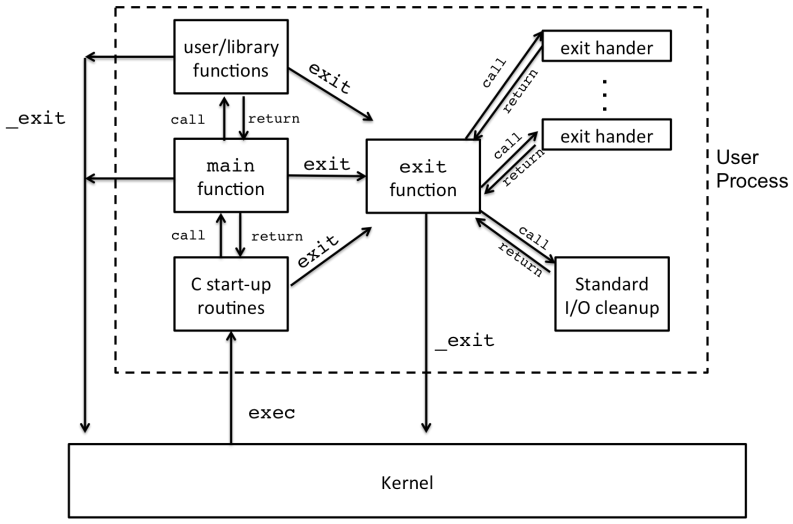
5/3/1/0 1. What is the difference between **_exit()** and **exit()** and **_Exit()**?

5/3/1/0 2. When a process returns from **main()** which of three different exit calls is actually used? What is the exit value?

10/8/4/0 3. In the diagram below, place a circle along the exit path for the following program:

```
void fun(){
     _exit(1);
}

int main(){
     exit(0);
     fun();
}
```




5/3/1/0 4. Match each of the IO buffering settings to their mode options in **setvbuf()**:

**_IONBF** _____ a) unbuffered

**_IOFBF** _____ b) line buffered

**_IOLBF** _____ c) fully buffered

5/3/1/0 5. What is the difference between line buffered and fully buffered?

__/30

NAME: _____

6. Consider the following programs, what are their output? And, **explain.**

5/3/1/0

```
int main(){
     fprintf(stdout, "Hello World!");
     return 0;
}
```

5/3/1/0

```
int main(){
     fprintf(stdout, "Hello World!");
     exit(0);
}
```

5/3/1/0

```
int main(){
     fprintf(stdout, "Hello World!");
     _Exit(0);
}
```

5/3/1/0

```
int main(){
     fprintf(stderr, "Hello World!");
     _exit(0);
}
```

5/3/1/0

7. Why does the following code snippet prperly check for a failed call to **execv()**?

```
int main(){
    char * ls_args[2] = { "/bin/ls", NULL} ;

    execv( ls_args[0], ls_args);
    perror("execve failed");

    _exit(1); //failure
}
```

5/3/1/0 8. Consider setting up an **argv** array to be passed to **execv()** for the execution of following command: **ls –l –a /bin /usr/bin** Fill in the argv decleartation:

**char * argv[] = {** **}**

__/30

NAME: _____

5/3/1/0 9. The **fork()** system call is the only function that returns twice when succesful. Explain this phenominom?

5/3/1/0 10. The typdefined type of a process identifier, or **pid**, is **pid_t**. What real type is a **pid_t**?

5/3/1/0 11. What system call is used to determine the current **pid** of a process? What system call is used to determine the parrent's process id of the calling process?

5/3/1/0 12. In the following small program, which program's pid would typicall be the parents for the output? Explain.
Assume the program is run from the shell like: **./print_ppid**

```
int main(){

    printf("Parent pid: %d\n",
            getppid());

}
```

5/3/1/0 13. The **wait()** system call waits for the status change of a child process: What is a typical status change that you could wait on?

5/3/1/0 14. Open the manual for **wait()**, match the status macro to its description:

| | | |
|---|---|---|
| WIFEXITED(status) | ____ | (a) Returns true if the child process was terminated by a signal |
| WIFEXITSTATUS(status) | ____ | (b) Returns true if the child terminated normally |
| WIFSIGNALED(status) | ____ | (c) Retrieves the exit staus of the child. |

__/30

NAME: _____

15. Assume you were writing a program that checked if a file existed by using **ls**. (This is a silly way to do this, but just for the sake of argument)

Recall that **ls** returns an exit status of 2 when the file **does not exist** and it cannot list it, and ls returns an exit status of 0 when the file does exist and can be listed.

Complete the **wait()** portion of the program below. The output should be **EXISTS!** if the file specified in argv[1] exists and **DOES NOT EXIST!** If the file specified in argv[1] does not exist. (hint: actually try and complete the program on your computer)

```
#include <unistd.h>
#include <stdlib.h>
#include <stdio.h>
#include <sys/wait.h>
#include <sys/types.h>

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

  pid_t cid;
  char * ls_args[] = {"ls", NULL, NULL};

  if(argc == 2){
    ls_args[1] = argv[1];
  }

  cid = fork();
  if( cid == 0 ){ /*child*/
    execvp(ls_args[0],ls_args);
    exit(1); /*error*/
  }
  /*parent*/
  int status;
  wait(&status);
```

10/8/6/3/0

__/10

```
}
```