Help Session 2

Programming Assignment 2

Outline

- Learn how to use the following
  - fork()
  - execlp()
  - wait()

Assignment Information

- Two executables will be needed
  - Coordinator– Main program, that creates 4 instances of Checker and passes the arguments to it
  - Checker– Checks whether the given argument 2(dividend) is divisible by argument 1(divisor)

Coordinator.c

- Will take exactly 5 command line arguments. First argument is the divisor and the rest are dividends
- Is responsible with launching 4 instances of the Checker program
- It will selectively pass the arguments to each instance of the Checker
- Should complete an entire cycle of fork(), exec() and wait() before going for the next process instance creation.

fork()

fork()

- Generates an exact copy of parent process except for the value it returns
- In a child process, fork() returns zero
- In the parent process it will return the child's process ID
- Any process can retrieve its process ID with getpid(), and its parent process ID with getppid()
- Include unistd.h
- Syntax:
  - pid_t fork();

wait()

- Makes Coordinator wait until the child has been entirely executed
- Use WIFEXITED() to check whether child process has completed its execution
- Use WEXITSTATUS() to retrieve return value of child process
- Syntax
  - pid_t wait(int* stat_loc);

execlp()

- Executes a new program within a child process
- Arguments passed are the name of the executable and file name like “./Checker”, “Checker.c”
- Also pass any needed command line arguments as parameters
- Terminate list of arguments with NULL
- Syntax
  - int execlp(“./Executable_name”, “Filename”, const char *arg, …, NULL);

```
# List of files
C_SRCS = Coordinator.c Checker.c
C_OBJS = Coordinator.o Checker.o
#C_HEADERS
OBJS = ${C_OBJS}
EXE1 = Coordinator
EXE2 = Checker
#Compiler and loader commands and flags
GCC = gcc
GCC_FLAGS = -std=c11 -g  -Wall -c -I.
LD_FLAGS = -std=c11 -g  -Wall -I.
#Compile .c files to .o files
.c.o:
	$(GCC) $(GCC_FLAGS) $<
#Target is the executable
all: Coordinator Checker
Coordinator: Coordinator.o
	$(GCC) $(LD_FLAGS) Coodinator.o -o $(EXE1)
Checker: Checker.o
	$(GCC) $(LD_FLAGS) Checker.o -o $(EXE2)
#Recompile C objects if headers change
$(C_OBJS): ${C_HEADERS}
#Clean up the directory
clean:
	rm -f *.o *~ $(EXE1) $(EXE2)
package:
	tar -cvf Lastname-Firstname-HW2.tar Coordinator.c Checker.c Makefile README.txt
```