CS 354 Fall 2025

Lab 2: Process Scheduling and Context Switching [20 pts]

Due: 9/15/2025 (Monday), 11:59 PM

1. Purpose

The objective of this lab assignment is to understand process priorities, how an operating system uses priorities to schedule processes, and how processes behave. You will have an opportunity to learn the effect of process priorities and how the operating system schedules processes that have equal priority.

---

2. Readings

Chapters 1, 2 from the XINU textbook.

---

3. Source code

Start from a fresh copy of XINU `xinu-fall2025.tar.gz` by running the command:

```
tar zxvf /homes/cs354/xinu-fall2025.tar.gz
```

---

4. Create and run processes that output characters

Please follow each step in this section in sequential order.

Note that some of the steps map to questions shown in Section 5. It may be a good idea to answer questions as you walk through each step.

4.1 Output priority of currently running process

The Xinu call

```
getpid()
```

returns the process ID of the running process, and the call

```
getprio(X)
```

returns the scheduling priority of the process with PID X. Add code in the beginning of `main()` (inside `main.c`) that uses `kprintf` to print the priority of the process running `main` (knowing the priority will help you answer questions).

4.2 Print a character n times

Add a function to `main.c` named

```
prch(char ch, int32 ntimes)
```

The function takes two arguments: a character `ch` to print and `ntimes` how many times to print the character. Have your function enter a loop, print the character `ch` `ntimes`, and then return. Use `kputc` to print the character (`kputc` takes a character as an argument).

To test your function, add the call:

```
prch('Z', 5);
```

to the main function, and run the result.

4.3 Create and run processes printing characters

Create and resume two processes that run `prch`. Name one of them `prA`, and the other `prB`. Have `prA` output 800 As and have `prB` output 800 Bs. Make `prA` have priority 10

and prB have priority 11. After creating and resuming the processes, have the main function use kprintf to print:

```
"\nmain is exiting\n\n"
```

and then return (i.e., exit). Run the result.

4.4 Check completion of processes

Change the code by adding a function compl that prints the identity of a process when the process exits. Modify the call to

```
resume(create... ));
```

so it saves the process ID in a global variable. Declare two global variables pidA and pidB, both of type pid32. Place the process ID of prA in pidA and the process ID of prB in pidB. Have compl take an argument pid that is a process ID. If pid matches pidA, use kprintf to print

```
"\n\nprocess prA has completed\n"
```

if pid matches pidB, have compl use kprintf to print

```
"\n\nprocess prB has completed\n"
```

Add two calls to compl at the end of main. Make each of them to use receive(), and then send the resulting message (by calling send()) to your function compl.

```
compl(receive());
```

4.5 Change priority of process printing characters

Modify your code to change the priority of prB from 11 to 21, and run the result.

4.6 Change priority of process printing characters (again)

Change the priority of `prB` to 10 (the same as `prA`), and run the result.

4.7 Add a third process that prints characters

Make the priority of both `prA` and `prB` to 12. After both have been created and resumed, insert code that delays 1 millisecond:

```
sleepms(1);
```

For now, all you need to know is that `sleepms(1)` will make the calling process ineligible to use the processor for 1 ms, make that process eligible after, and then reschedule if needed. Follow the call to `sleepms` with code that creates and resumes a third process, `prC`. Give `prC` a priority of 15 and have it print 'C' 100 times. Also, modify `compl` to print a message about `prC` completing, and add a third call to `compl` at the end of main. Run the result.

---

5. Questions

Place **short answers** to the following questions in a file named `questions.pdf`. In case of using document software (e.g., Word), please make sure to convert the file to PDF.

1. In Section 4.3, in what order did processes complete? Explain exactly how and when the scheduler chose to switch from one process to another.

2. In Section 4.4, in what order did processes complete? Explain how and when the scheduler chose to switch from one process to another.

3. In Section 4.5, in what order did processes complete? Explain how the scheduler made decisions to change from one process to another.

4. In Section 4.6, in what order did processes complete? Explain how the scheduler made decisions to change from one process to another.

5. In Section 4.7, state the sequence of events that occur, including process exit, and explain how the resulting output is affected.

---

6. Submission

General instructions:

When implementing code in the labs, please maintain separate versions/copies of code so that mistakes such as unintentional overwriting or deletion of code is prevented. This is in addition to the efficiency that such organization provides. You may use any number of version control systems such as GIT and RCS. Please make sure that your code is protected from public access. For example, when using GIT, use git that manages code locally instead of its on-line counterpart github. If you prefer not to use version control tools, you may just use manual copy to keep track of different versions required for development and testing. More vigilance and discipline may be required when doing so.

You must submit a directory named `lab2` that contains both required files:

- Your final version of `main.c` from Xinu
- `questions.pdf` containing your answers

Go to the directory where `lab2` is a subdirectory.

For example, if `/homes/alice/cs354/lab2` is your directory structure, go to `/homes/alice/cs354`.

Type the following command to submit the directory with `turnin`:

```
turnin -c cs354 -p lab2 lab2
```

Be sure the files inside the directory are named exactly `main.c` and `questions.pdf`.

You can check/list the submitted files using

```
turnin -c cs354 -p lab2 -v
```