SER 334 A Session

SI Session

Thursday, February 1st 2024

7:00 pm - 8:00 pm MST

Agenda

- Process Tracing
- Parallelism
- Threading Issues
- Dining Philosophers

SI Session Expectations

Thanks for coming to the **SER 334** SI session. We have a packed agenda and we are going to try to get through as many of our planned example problems as possible. This session will be recorded and shared with others.

- If after this you want to see additional examples, please visit the drop-in tutoring center.
- We will post the link in the chat now and at the end of the session.
  - tutoring.asu.edu
- Please keep in mind we are recording this session and it will be made available for you to review 24-48 hours after this session concludes.
- Finally, please be respectful to each other during the session.

Interact with us:

Zoom Features

Zoom Chat

- Use the chat feature to interact with the presenter and respond to presenter's questions.
- Annotations are encouraged

Module 5 Sample

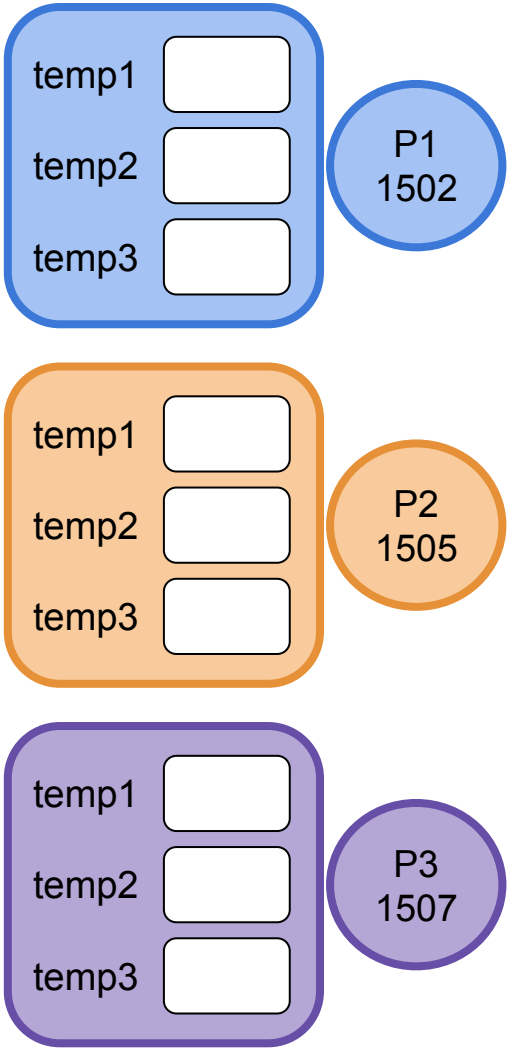
SER 334

Module 5 Sample

4. [Lisonbee] Trace the program below, identify the values of the pids at lines A, B, C, D, E, and F. (Assume that the actual pid of Process 1 is 1502, Process 2 is 1505, and Process 3 is 1507. Also assume that fork will always succeed.) [4 points]

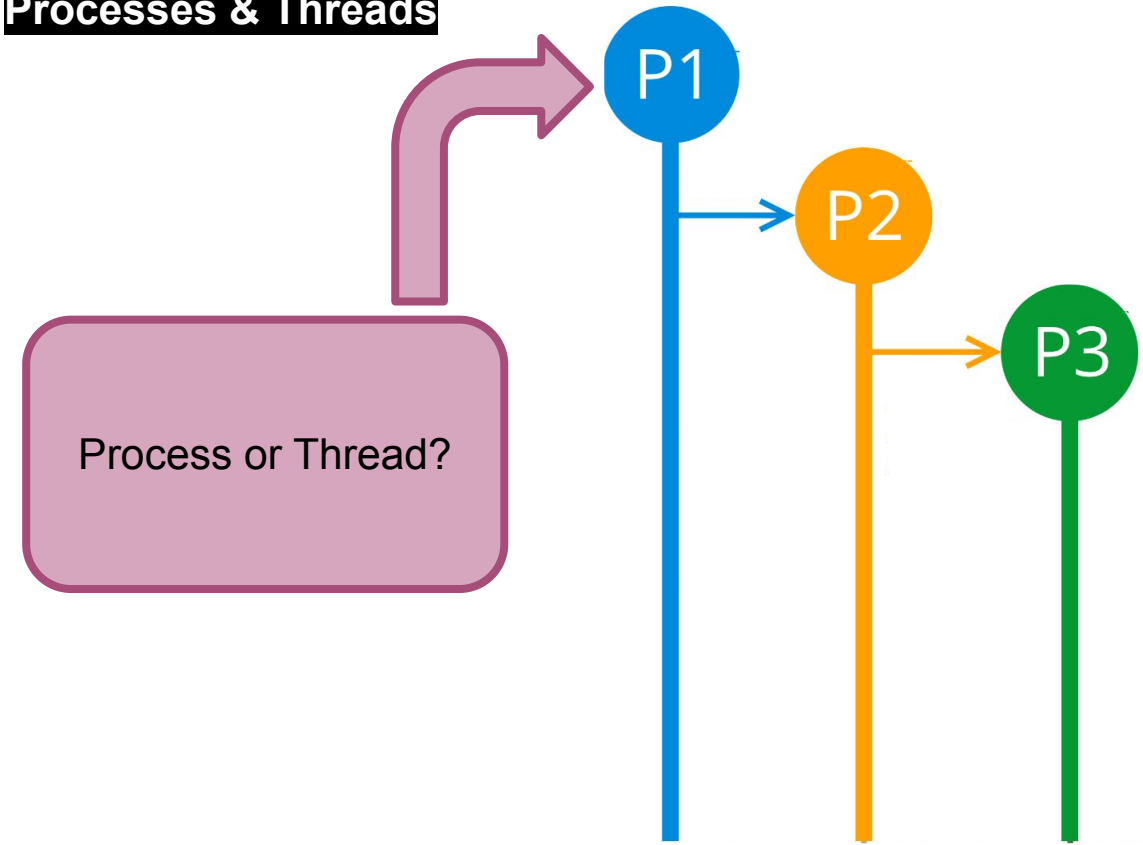
A	
B	
C	
D	
E	
F	

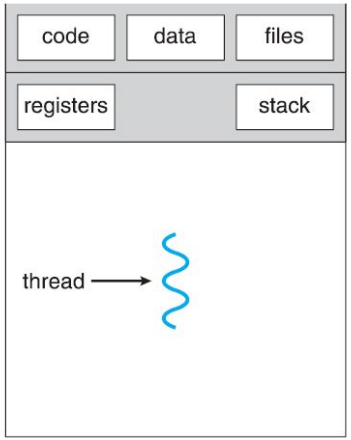
```
int main() {
    pid_t temp1, temp2, temp3;
    temp1 = fork();

    if (temp1 < 0) { /* Error occurred */
        fprintf(stderr, "Fork Failed");
        return 1;
    }
    else if (temp1 == 0) { /* Process 2 */
        temp2 = fork();

        if (temp2 < 0) { /* Error occurred */
            fprintf(stderr, "Fork Failed");
            return 1;
        }
        else if (temp2 == 0) { /* Process 3 */
            temp3 = getpid();
            printf("temp2 = %d", temp2); /* A */
            printf("temp3 = %d", temp3); /* B */
        }
        else { /* Process 2 */
            temp3 = getpid();
            printf("temp2 = %d", temp2); /* C */
            printf("temp3 = %d", temp3); /* D */
            wait(NULL);
        }
    }
    else { /* Process 1 */
        temp2 = getpid();
        printf("temp1 = %d", temp1); /* E */
        printf("temp2 = %d", temp2); /* F */
        wait(NULL);
    }
    return 0;
}
```



SER 334

Processes & Threads





single-threaded process

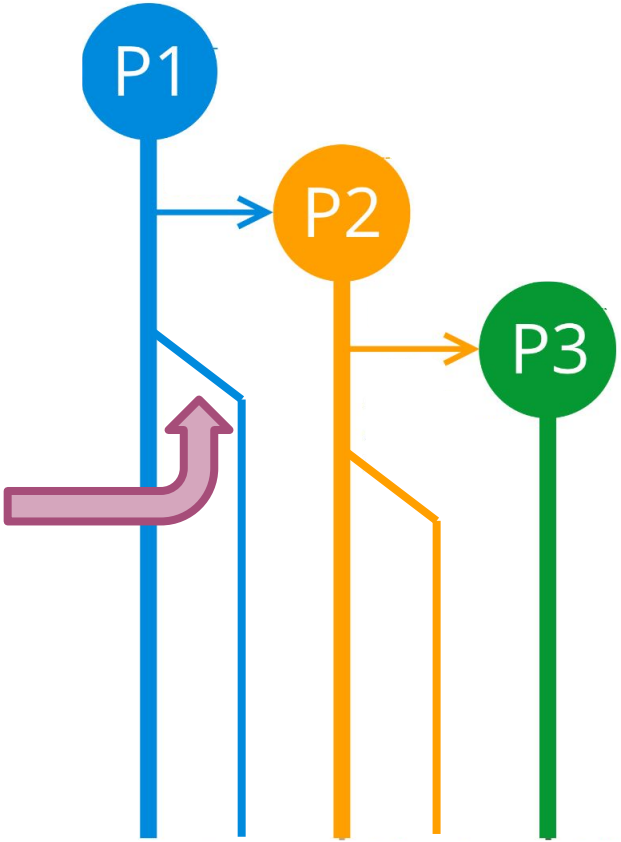
SER 334

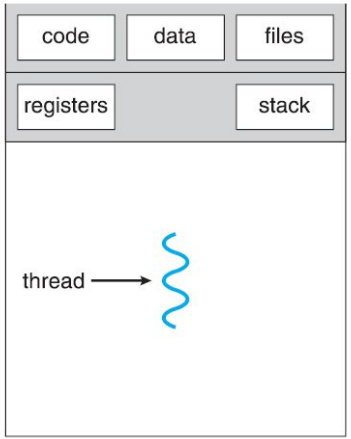
Depicting Threads

You get a stack!

You get a stack!

pthread_create

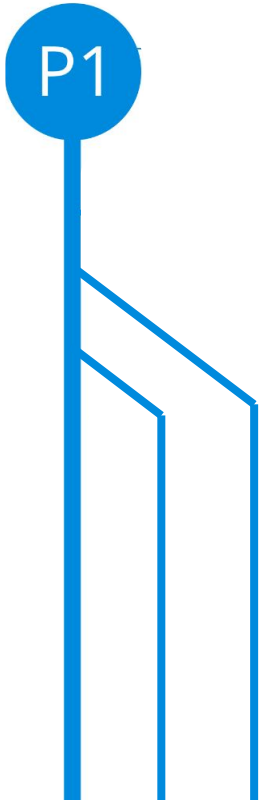


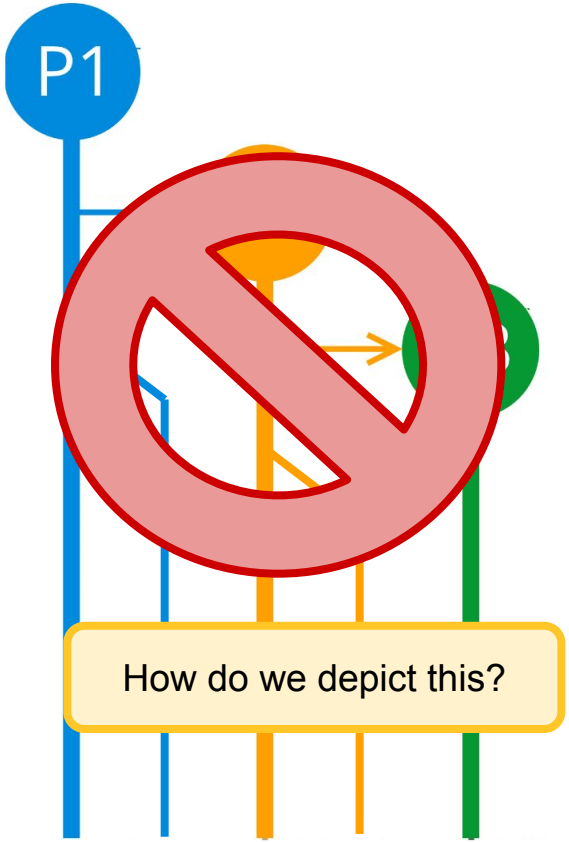


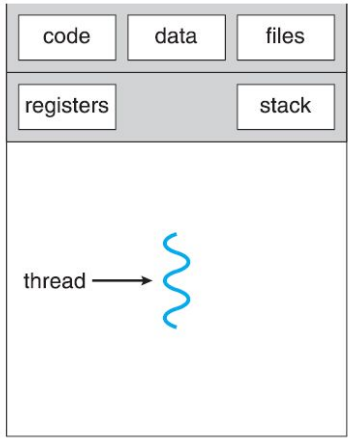
single-threaded process

SER 334

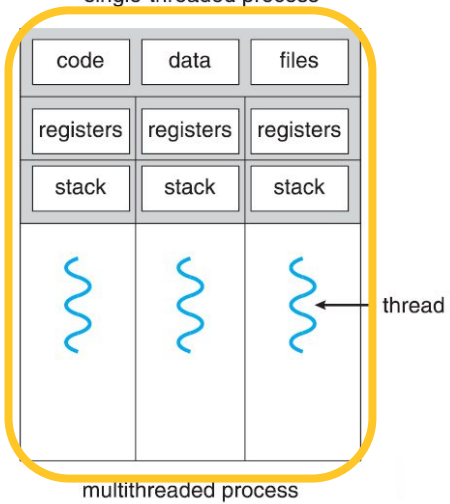
Depicting Threads







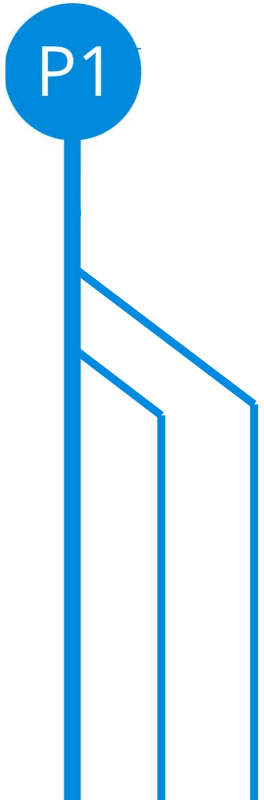
single-threaded process

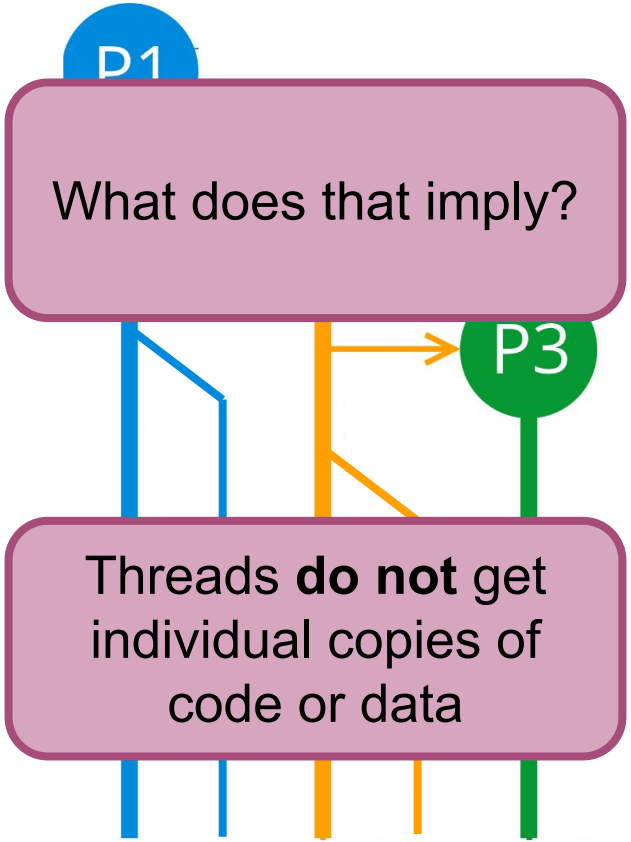


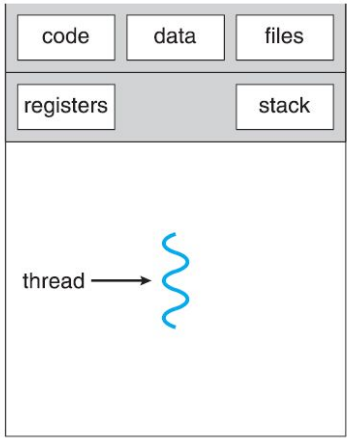
multithreaded process

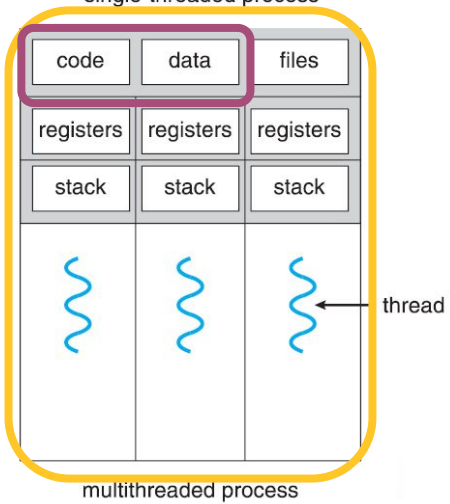
SER 334

Depicting Threads









SER 334

Module 6 Samples

4. [Silberschatz 4.15 edited] Consider the following code fragment and answer the following questions:

(a) Using “lifeline notation”, draw the creation of processes and threads during execution.
(b) How many unique processes are created? (Do not include the initial process.)
(c) How many unique threads are created? (Hint: processes don't count!)

```
pid_t pid;
temp = fork();

if (temp == 0) {
        fork();
}

thread_create(...);
```

SER 334
Multi-Threading

What's the difference??

Foreshadowing for CPU scheduling!

Parallelism

vs.

Concurrency

Simultaneous Execution

Near-Time Execution

***Run* at the same time**

***Exist* at the same time**

SER 334

Parallelism Types

Data Parallelism

vs.

Task Parallelism

Same Task
Different Data

Same Data
Different Task

SER 334

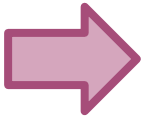
Parallelism Types

Data Parallelism

vs.

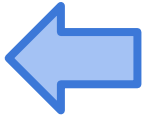
Task Parallelism

Many data sets



Same Task
Different Data

Same Data
Different Task



One data set

Check out the recording for the solution!

SER 334
Threading Issues

- Test and Debugging
- Data Splitting
- Identifying Tasks
- Data Dependency
- Balance

A. Identifying independent functionality

B. Ensuring comparable amounts of work

C. Partitioning and minimizing memory use

D. Some tasks must be performed sequentially

E. Multiple “executors” for each line makes it harder to identify the culprit

Module 5 Sample

SER 334

Enter topic here

6. [Lisonbee] Consider a system where two processes (a producer and a consumer) use a message-passing system to communicate, and each process does work at different rates. The producer can produce (perform work) at any rate, but the consumer has to wait for the producer to complete its task before moving on. Based on this system's needs, explain whether a synchronous or asynchronous communication system would be a better choice and why. [2 points]

Synchronous

Asynchronous

SER 334

Scratch Space

Upcoming Events

SI Sessions:

- Sunday, February 4th at 7:00 pm MST
- Monday, February 5th at 7:00 pm MST

- ~~Sunday, February 11th at 7:00 pm MST~~ **Cancelled - Good luck on Exam 2!**

Review Sessions:

- Exam 2 Review: Thursday, February 8th 7:00 pm - 9:00 pm MST
- Exam 3 Review: TBD

Questions?

Survey:

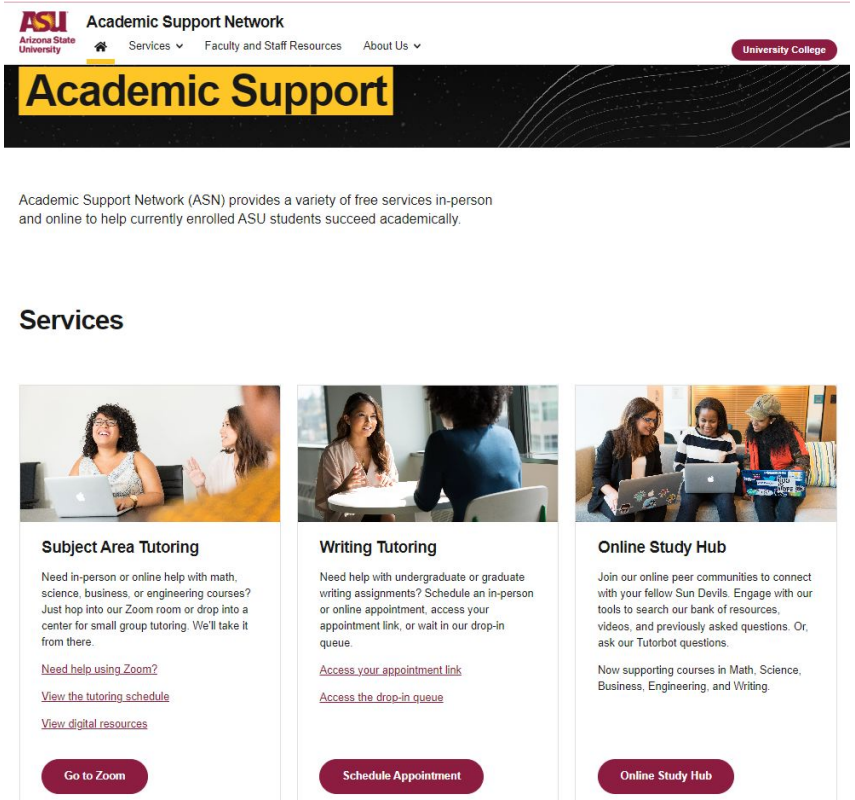
http://bit.ly/ASN2324

18

More Questions?

Check out our other resources!

tutoring.asu.edu

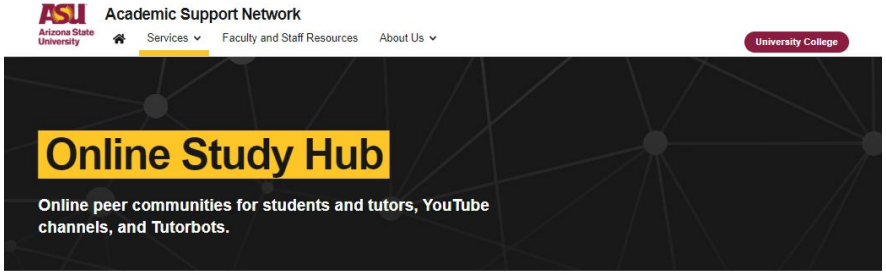


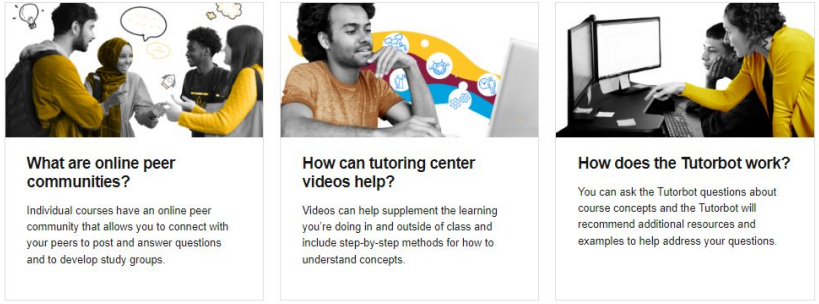
1. **Click on ‘Go to Zoom’ to log onto our Online Tutoring Center.**
2. **Click on ‘View the tutoring schedule’ to see when tutors are available for specific courses.**

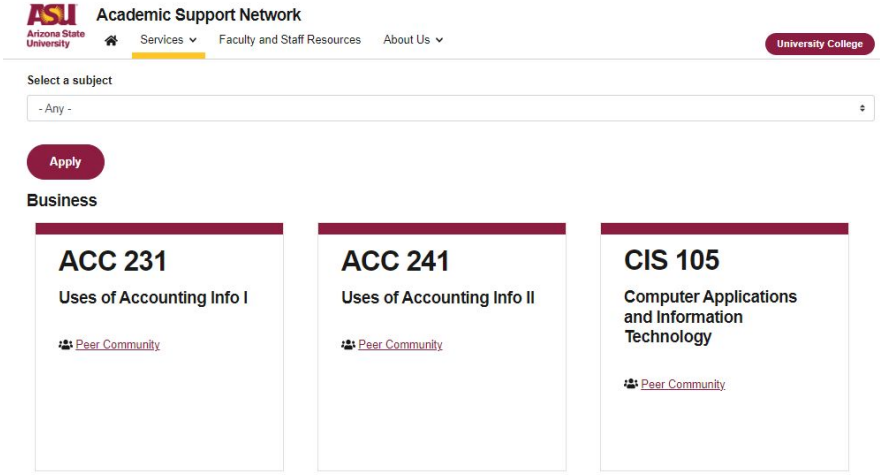
More Questions?

Check out our other resources!

tutoring.asu.edu/online-study-hub







Don't forget to check out the Online Study Hub for additional resources!

Additional Resources

- Course Repo
- Course Discord
- BMP File Format (Wiki)
- Linux Kernel API
- Bootlin - Linux Cross Referencer
- Dining Philosophers Interactive