

SER 321 A Session

SI Session

Thursday, February 6th 2025

7:00 pm - 8:00 pm MST

Agenda



Protobuf PSA

Threaded Pitfalls

Dining Philosophers

Traffic Analogy

Concurrency Structures

Threading the Server

SI Session Expectations

Thanks for coming to the **SER 321** 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

Quick PSA for Protobufs!

Make sure you watch all the lecture videos!

You must generate the Protobufs for use!

Check out the recording for the discussion!

4-2 Starter Code

SER 321

Protobufs

Option 1:
IDE

The screenshot shows an IDE with the following components:

- Project View (Left):** Shows the project structure for 'Assignment4.2given'. The 'src/main/java/server' directory is selected, containing 'SockBaseServer'.
- Code Editor (Center):** Displays the 'SockBaseServer.java' file. The code includes package declarations, imports, and a class definition for 'SockBaseServer' with various static variables and methods.
- Gradle View (Right):** Shows the 'Tasks' section for 'Assignment4.2given'. A yellow arrow points from the 'generateProto' task in the 'other' section to the code editor.

Code Snippet (SockBaseServer.java):

```
package server;

import ...

class SockBaseServer {
    static String logFilename = "logs.txt"; 3 usages

    // Please use these as given so it works with our test cases
    static String menuOptions = "\nWhat would you like to do? \n 1 - ...
    static String gameOptions = "\nChoose an action: \n (1-9) - ...

    ServerSocket serv = null; no usages
    InputStream in = null; 3 usages
    OutputStream out = null; 4 usages
    Socket clientSocket = null; 4 usages
    private final int id; // client id 2 usages

    Game game; // current game 3 usages

    private boolean inGame = false; // a game was started (you can ...
    private String name; // player name 4 usages

    private int currentState = 1; // I used something like this to ...

    private static boolean grading = true; // if the grading board ...

    public SockBaseServer(Socket sock, Game game, int id) { 1 usage
        this.clientSocket = sock;
        this.game = game;
        this.id = id;
    }
}
```

Gradle Tasks:

- build
- build setup
- documentation
- help
- other
 - arguments
 - compileJava
 - compileTestJava
 - components
 - dependentComponents
 - extractIncludeProto
 - extractIncludeTestProto
 - extractProto
 - extractTestProto
 - generateProto
 - generateTestProto
 - model
 - prepareKotlinBuildScriptModel
 - processResources
 - processTestResources
 - runClient
 - runServer
 - runServerGrading
- verification
- Dependencies

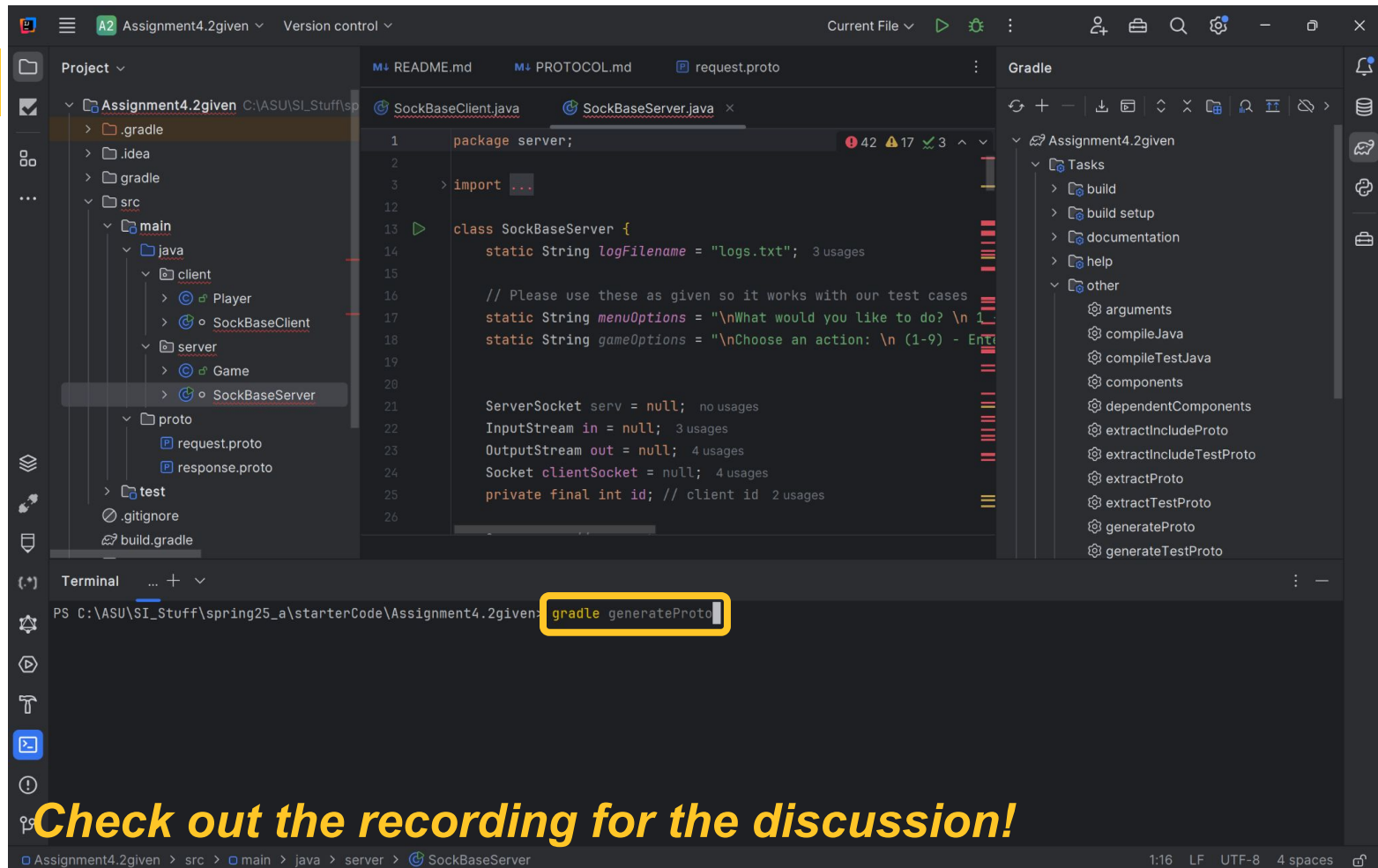
Check out the recording for the discussion!

4-2 Starter Code

SER 321

Protobufs

Option 2:
Command
Line



```
package server;

import ...

class SockBaseServer {
    static String logFilename = "logs.txt"; 3 usages

    // Please use these as given so it works with our test cases
    static String menuOptions = "\nWhat would you like to do? \n 1 - ...
    static String gameOptions = "\nChoose an action: \n (1-9) - ...

    ServerSocket serv = null; no usages
    InputStream in = null; 3 usages
    OutputStream out = null; 4 usages
    Socket clientSocket = null; 4 usages
    private final int id; // client id 2 usages
```

Terminal

```
PS C:\ASU\SI_Stuff\spring25_a\starterCode\Assignment4.2given> gradle generateProto
```

Assignment4.2given > src > main > java > server > SockBaseServer

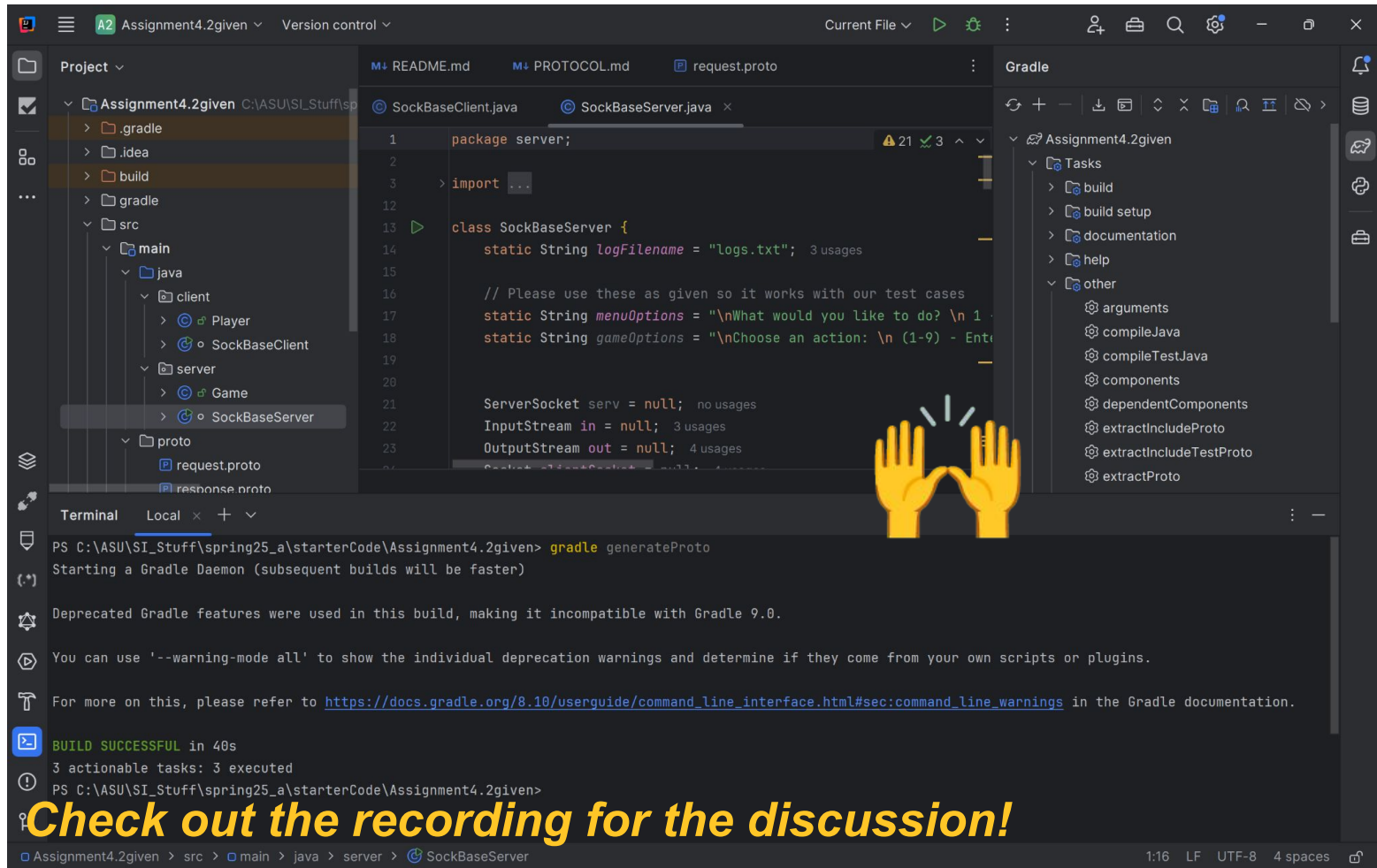
1:16 LF UTF-8 4 spaces

Check out the recording for the discussion!

4-2 Starter Code

SER 321

Protobufs



SER 321

Protobufs

Options for Message Creation

```
Response.newBuilder()  
    .setResponseType(Response.ResponseType.GREETING)  
    .setMessage("Hello " + name + " and welcome to a simple game of Sudoku.")  
    .setMenuoptions(menuOptions)  
    .setNext(currentState)  
    .build();
```

SockBaseServer

Create the message in
a single statement

Create the message in
increments

```
Request.Builder req = Request.newBuilder();  
  
switch (response.getResponse_type()) {  
    case GREETING:  
        System.out.println(response.getMessage());  
        req = chooseMenu(req, response);  
        break;
```

SockBaseClient - main

SER 321**Protobufs**

Options for Message Creation

Need one
more step...

```

static Request.Builder chooseMenu(Request.Builder req, Response response) throws IOException {
    while (true) {
        System.out.println(response.getMenuoptions());
        System.out.print("Enter a number 1-3: ");
        BufferedReader stdin = new BufferedReader(new InputStreamReader(System.in));
        String menu_select = stdin.readLine();
        System.out.println(menu_select);
        switch (menu_select) {
            // needs to include the other requests
            case "2":
                // this is not a complete START request!! Just as example
                req.setOperationType(Request.OperationType.START);
                return req;
            default:
                System.out.println("\nNot a valid choice, please try again.");
                break;
        }
    }
}

```

SockBaseClient - chooseMenu

```

Request.Builder req = Request.newBuilder();

switch (response.getResponseCode()) {
    case 200:
        System.out.println(response.getMessage());
        req = chooseMenu(req, response);
        break;
}

```

SockBaseClient - main

```
req.build().writeDelimitedTo(out);
```

Check out the recording for the discussion!

SER 321

Protobufs

Parsing Messages

GETTERS!

```
System.out.println("Got a response: " + response.toString());
```

```
switch (response.getResponse_type()) {  
    case GREETING:  
        System.out.println(response.getMessage());  
        req = chooseMenu(req, response);  
        break;
```

Fetch a single value

Fetch a *repeated* value

```
for (Entry lead: response3.getLeaderList()){  
    System.out.println(lead.getName() + ": " + lead.getPoints());  
}
```

Check out the recording for the discussion!

SER 321

System Layout

You have two systems...

How can we test our server with multiple clients?



?



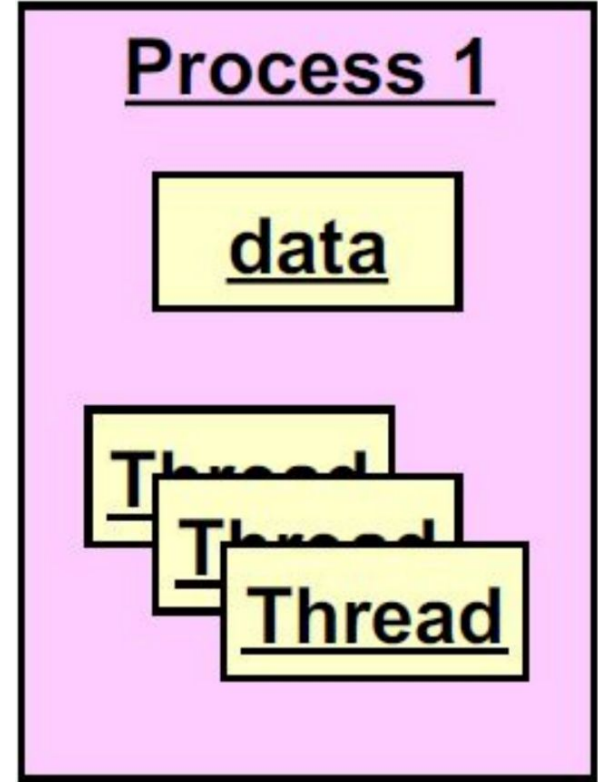
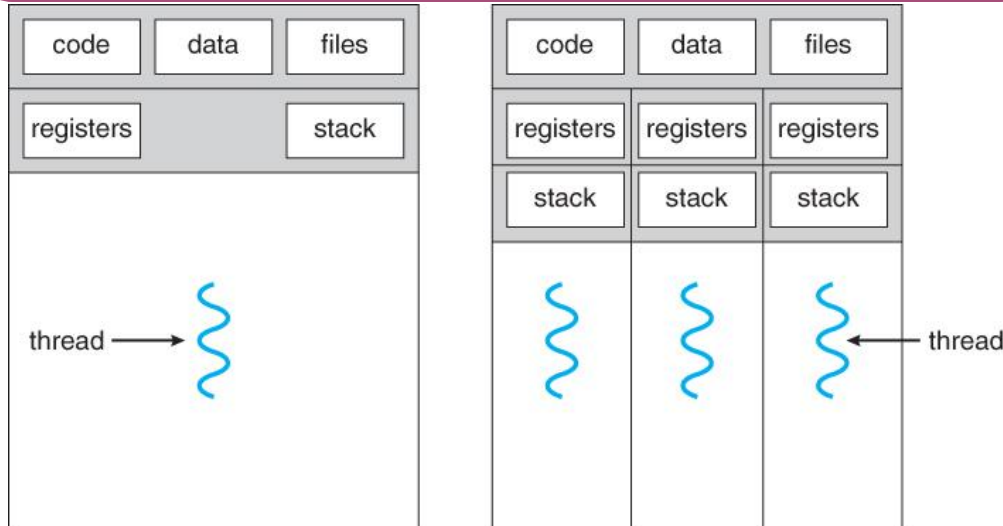
Check out the recording for the discussion!

SER 321

Threads

What does that imply?

Remember that they exist
within the parent process



SER 321

Threading Pitfalls

Check out the recording for the solution!

Race Condition

A thread never gains access to the resource it needs

Starvation

A thread is only able to acquire some of the resources it needs

Deadlock

More than one thread accesses a single resource at the same time

SER 321

Threading Pitfalls

As the project name implies, we encounter a **deadlock**.

But what happened?

```
class SockClient {
    public static void main (String args[]) throws Exception {
        Socket      sock = new Socket( host: "localhost", port: 8888); //Any IP name

        ObjectInputStream in = new ObjectInputStream(sock.getInputStream());
        ObjectOutputStream out = new ObjectOutputStream(sock.getOutputStream());

        String s = (String) in.readObject();
        out.writeObject("Back at you");

        in.close();
        out.close();
        sock.close();
    }
}
```

Client

```
class SockServer {
    public static void main (String args[]) throws Exception {

        int count = 0;
        ServerSocket  serv = new ServerSocket( port: 8888);

        Socket sock = serv.accept();

        ObjectInputStream in = new ObjectInputStream(sock.getInputStream());
        ObjectOutputStream out = new ObjectOutputStream(sock.getOutputStream());

        String s = (String) in.readObject();
        System.out.println("Received " + s);
        out.writeObject("Back at you");
        System.out.println("Received " + s);

        in.close();
        out.close();
        sock.close();
    }
}
```

Server

Check out the recording for the discussion!

```
PS C:\ASU\SER321\examples_repo\ser321examples\Threads\NetworkDeadlock> gradle
server
<=====--> 75% EXECUTING [1m 33s]
> :server
█
```

```
PS C:\ASU\SER321\examples_repo\ser321examples\Threads\NetworkDeadlock> gradle
client
Starting a Gradle Daemon, 1 busy and 1 stopped Daemons could not be reused, use
--status for details
<=====--> 75% EXECUTING [53s]
> :client
█
```

SER 321

Threading Pitfalls

What does *Spaghetti Consumed* represent?

What does *Thinking* represent?

What does *Hungry* represent?

Check out the recording for the solution and discussion!

powered by NetLogo

Dining Philosophers

File: [New](#) [Revert to Original](#)
Export: [NetLogo](#) [HTML](#)

Mode: Interactive Commands and Code: Bottom

model speed

ticks: 6712

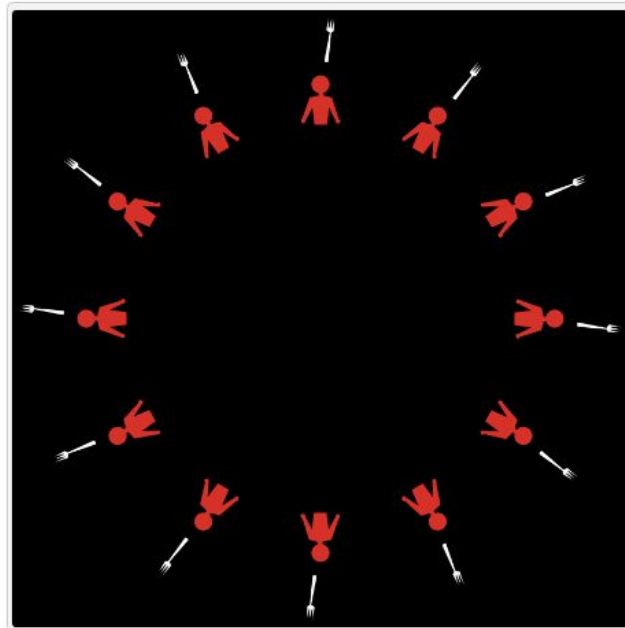
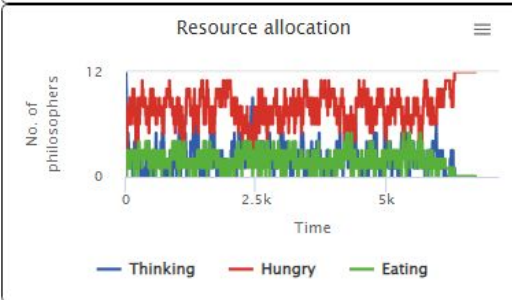
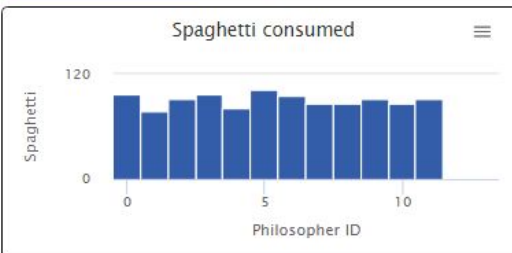
num-philosophers 12

setup go go once

hungry-chance 0.5

full-chance 0.5

☐ cooperation?

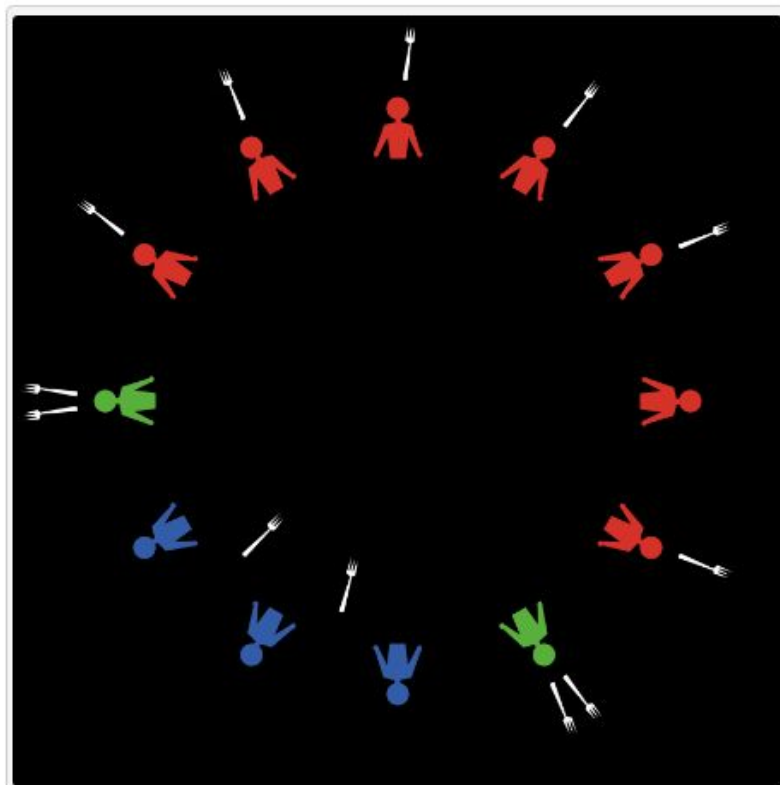


Check out the recording for the discussion!

SER 321

Threading Pitfalls

Can we take a guess at what is happening here?



SER 321

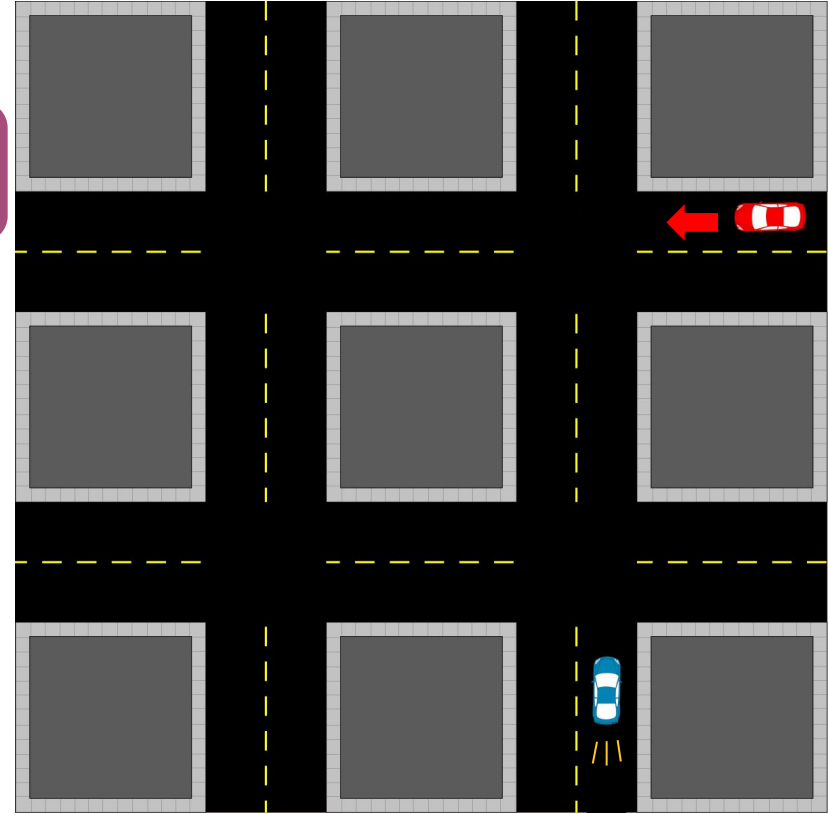
Threading Pitfalls

Check out the recording for the discussion!

Race Condition

Crash

More than one thread accesses a single resource at once



SER 321

Threading Pitfalls

Check out the recording for the discussion!

Race Condition

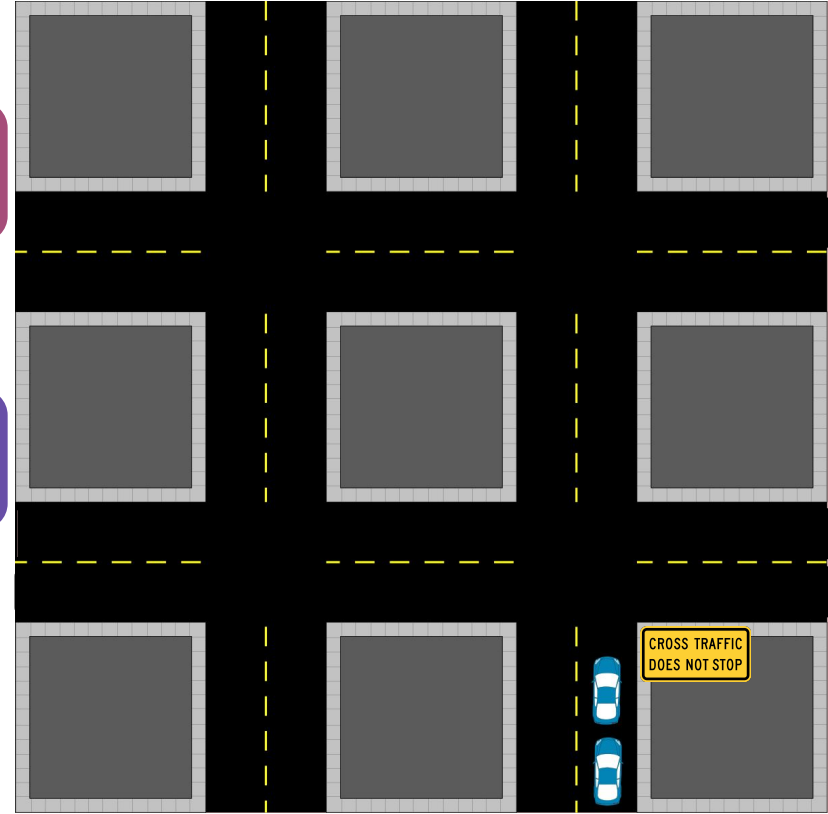
Crash

More than one thread accesses a single resource at once

Starvation

Cross Traffic

A thread never gains access to the resource it needs



SER 321

Threading Pitfalls

Check out the recording for the discussion!

Race Condition

Crash

More than one thread accesses a single resource at once

Starvation

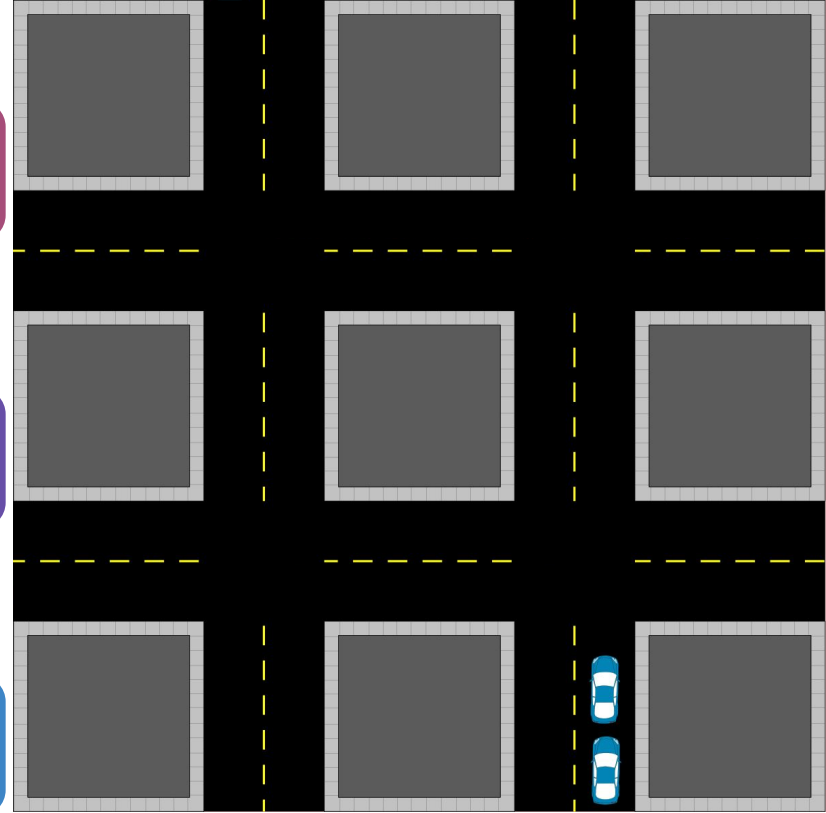
Cross Traffic

A thread never gains access to the resource it needs

Deadlock

Gridlock

A thread is only able to acquire some of the needed resources



SER 321

Concurrency Structures

Check out the recording for the discussion!

Can we name some concurrency structures?

Atomic Operations &
Variables

Locks

Semaphores

Monitors

Check out the recording for the discussion!

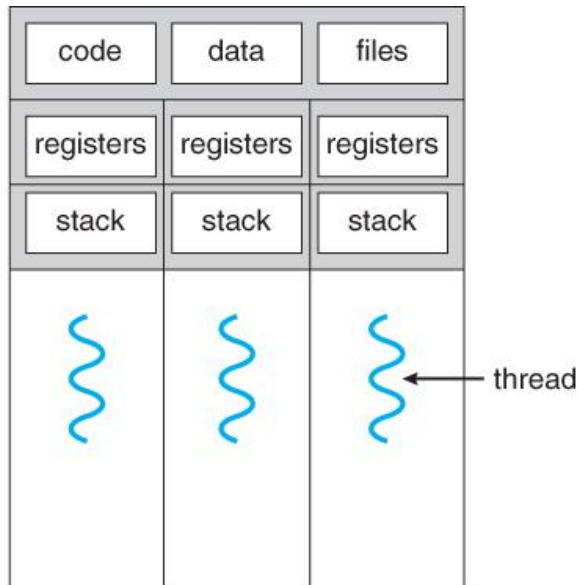
SER 321

Concurrency Structures

Atomic Operations & Variables

Recall *registers...*

Ensures updates are immediately visible for the local copy in *each thread*



main:

```
pushq    %rbp
movq     %rsp, %rbp
subq     $48, %rsp
call     __main
movl     $5, -4(%rbp)
movl     $12, -8(%rbp)
movl     -4(%rbp), %eax
addl     $7, %eax
movl     %eax, -12(%rbp)
movl     -8(%rbp), %edx
movl     -12(%rbp), %eax
addl     %edx, %eax
movl     %eax, -16(%rbp)
movl     -16(%rbp), %eax
movl     %eax, %edx
leaq     .LC0(%rip), %rax
movq     %rax, %rcx
call     printf
movl     $0, %eax
addq     $48, %rsp
popq     %rbp
ret
```

Check out the recording for the discussion!

SER 321

Concurrency Structures

Pros and Cons?



Locks

Acquire the Lock



Open & Enter

Close & Lock

Release the Lock

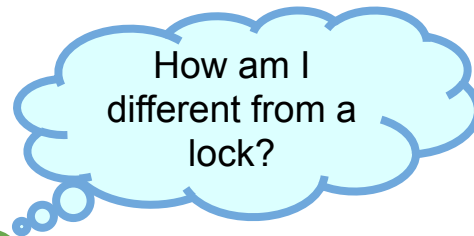


Unlock & Exit

Check out the recording for the discussion!

SER 321

Concurrency Structures



Semaphores

More
than one
stall!

Acquire Lock



Open & Enter

Close & Lock

Release Lock



Unlock & Exit

Semaphores support
more than one acquirer

When would that be beneficial?

Check out the recording for the discussion!

SER 321

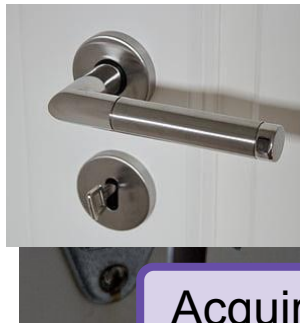
Concurrency Structures

Pros and Cons?

Monitors



You lock
the main
door
instead!



Acquire Lock



Open & Enter

Close & Lock

Release Lock



Unlock & Exit

Covers the
entire object

Check out the recording for the discussion!

SER 321

Concurrency Structures

RECAP

Atomic Operations &
Variables

YOU control the
locks directly

Locks

YOU control the
locks directly

Semaphores

YOU control the
locks directly

Monitors

Locks managed
for you

SER 321

Scratch Space

Upcoming Events

SI Sessions:

- Sunday, February 9th at 7:00 pm MST
- Tuesday, February 11th at 11:00 am MST
- Thursday, February 13th at 7:00 pm MST

Review Sessions:

- Tuesday, February 25th at 11:00 am MST - **Q&A Session**
- Thursday, February 27th at 7:00 pm MST - **Exam Review Session (2hrs)**

Questions?

Survey:

<https://asuasn.info/ASNSurvey>



More Questions?

Check out our other resources!

tutoring.asu.edu



Academic Support

Academic Support Network (ASN) provides a variety of free services in-person and online to help currently enrolled ASU students succeed academically.

Services



Subject Area Tutoring

Need in-person or online help with math, science, business, or engineering courses? Just hop into our Zoom room or drop into a center for small group tutoring. We'll take it from there.

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)

Go to Zoom



Writing Tutoring

Need help with undergraduate or graduate writing assignments? Schedule an in-person or online appointment, access your appointment link, or wait in our drop-in queue.

[Access your appointment link](#)

[Access the drop-in queue](#)

Schedule Appointment



Online Study Hub

Join our online peer communities to connect with your fellow Sun Devils. Engage with our tools to search our bank of resources, videos, and previously asked questions. Or, ask our Tutorbot questions.

Now supporting courses in Math, Science, Business, Engineering, and Writing.

Online Study Hub

1-

Go to Zoom

2-

[Need help using Zoom?](#)

[View the tutoring schedule](#)

[View digital resources](#)



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

 **Academic Support Network**

Services Faculty and Staff Resources About Us

University College

Online Study Hub

Online peer communities for students and tutors, YouTube channels, and Tutorbots.



What are online peer communities?

Individual courses have an online peer community that allows you to connect with your peers to post and answer questions and to develop study groups.



How can tutoring center videos help?

Videos can help supplement the learning you're doing in and outside of class and include step-by-step methods for how to understand concepts.



How does the Tutorbot work?

You can ask the Tutorbot questions about course concepts and the Tutorbot will recommend additional resources and examples to help address your questions.

Select a subject

- Any -

Apply



Academic Support Network



Services

Faculty and Staff Resources

About Us

University College

Select a subject

- Any -

Apply

Business

ACC 231

Uses of Accounting Info I

Peer Community

ACC 241

Uses of Accounting Info II

Peer Community

CIS 105

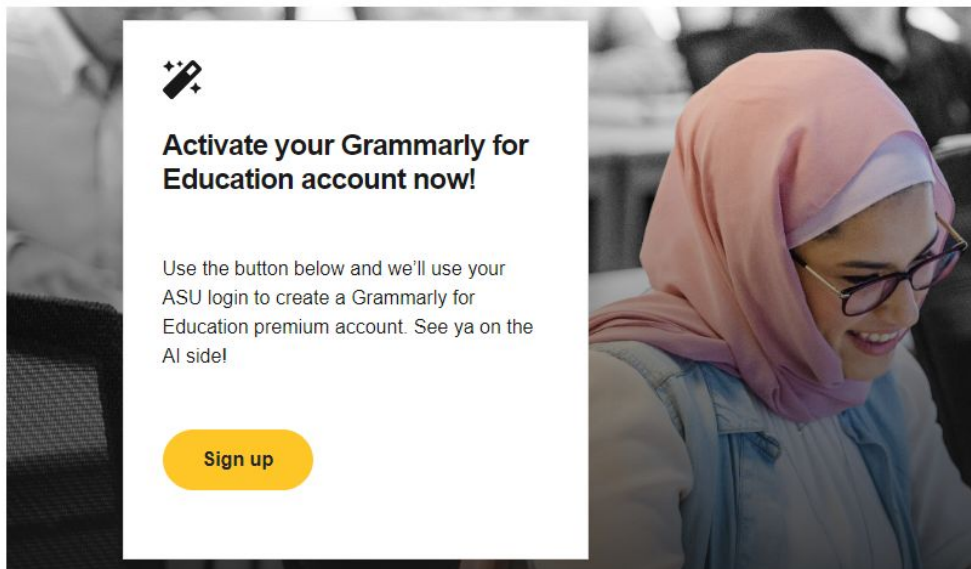
Computer Applications and Information Technology

Peer Community

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

Expanded Writing Support Available

Including Grammarly for Education, at no cost!



tutoring.asu.edu/expanded-writing-support

*Available slots for this pilot are limited

Additional Resources

- [Course Repo](#)
- [Gradle Documentation](#)
- [GitHub SSH Help](#)
- [Linux Man Pages](#)
- [OSI Interactive](#)
- [MDN HTTP Docs](#)
 - [Requests](#)
 - [Responses](#)
- [JSON Guide](#)
- [org.json Docs](#)
- [javax.swing package API](#)
- [Swing Tutorials](#)
- [Dining Philosophers Interactive](#)
- [Austin G Walters Traffic Comparison](#)