

SER 321 B Session

SI Session

Thursday, April 10th 2025

7:00 pm - 8:00 pm MST

Agenda



Protocol Buffers

Threading

Intro, Usage, and Pitfalls

Threading our System

Concurrency Structures

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

SER 321

Protobufs

***Check out
the
recording
for the
discussion!***

Quick PSA for Protobufs!

Make sure you watch all the [lecture videos](#)!

You must generate the Protobufs for use!

Sanity Check - what *are* Protocol Buffers?

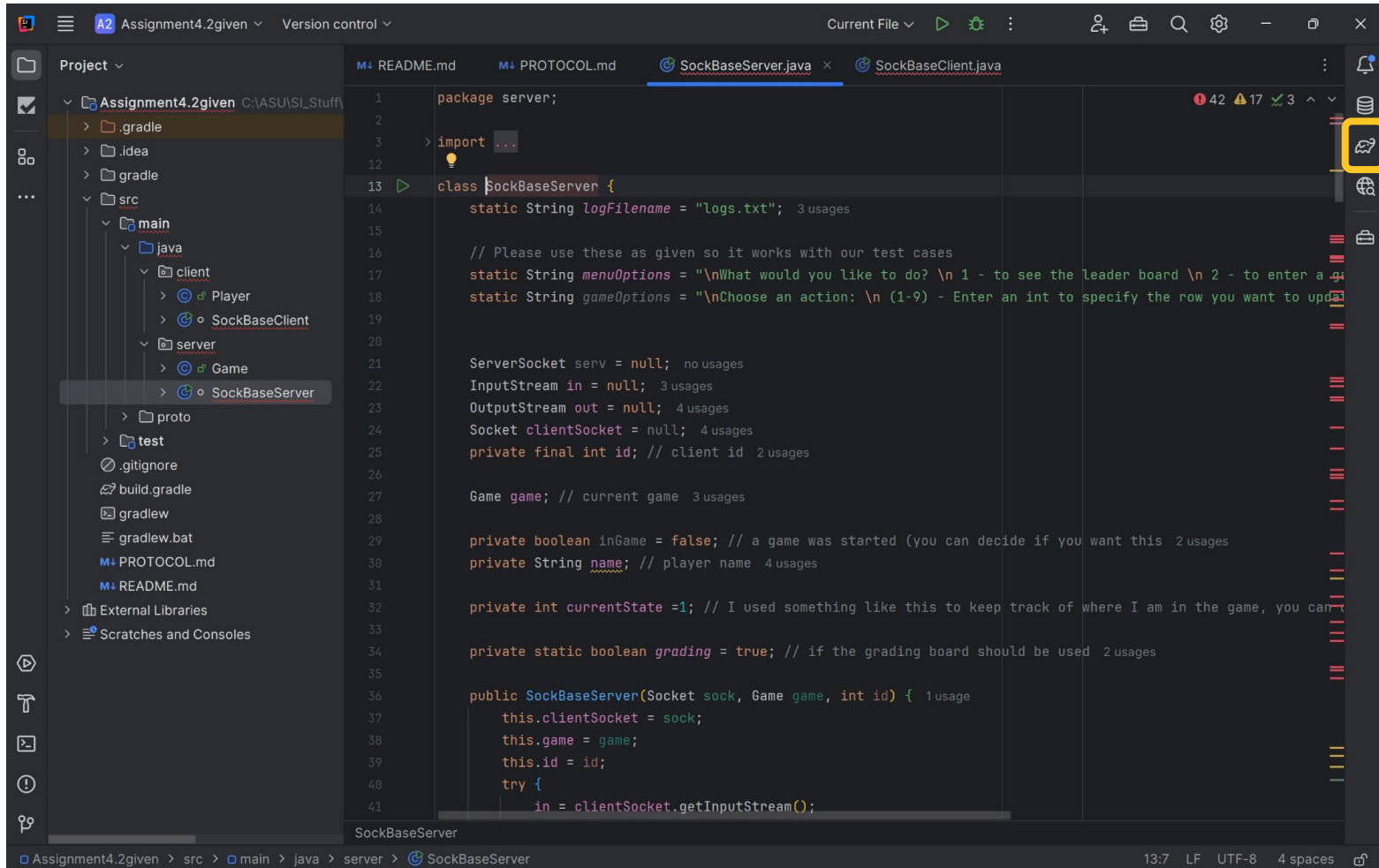
4-2 Starter Code

SER 321

Protobufs

Option 1:
IDE

**Check out
the
recording
for the
discussion!**



```
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 - to see the leader board \n 2 - to enter a game \n 3 - to quit \n";
    static String gameOptions = "\nChoose an action: \n (1-9) - Enter an int to specify the row you want to update \n";

    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 decide if you want this 2 usages)
    private String name; // player name 4 usages

    private int currentState = 1; // I used something like this to keep track of where I am in the game, you can use whatever you want
    private static boolean grading = true; // if the grading board should be used 2 usages

    public SockBaseServer(Socket sock, Game game, int id) { 1 usage
        this.clientSocket = sock;
        this.game = game;
        this.id = id;
        try {
            in = clientSocket.getInputStream();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

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

13:7 LF UTF-8 4 spaces

4-2 Starter Code

SER 321

Protobufs

Option 1:
IDE

**Check out
the
recording
for the
discussion!**

The screenshot shows an IDE with the following components:

- Project Explorer (Left):** Shows the project structure for 'Assignment4.2given'. The 'src' directory is expanded, showing 'main' > 'java' > 'server' > 'SockBaseServer'.
- Code Editor (Center):** Displays the content of 'SockBaseServer.java'. The code includes package declarations, imports, and a class definition for 'SockBaseServer' with various static variables and methods.
- Gradle Panel (Right):** Shows the 'Tasks' section for 'Assignment4.2given'. The 'generateProto' task is highlighted with a yellow arrow.

```
package server;

import ...

class SockBaseServer {
    static String logFilename = "logs.txt";
    // Please use these as given so it works with our test cases
    static String menuOptions = "\nWhat would you like to do? \n 1 - to see
    static String gameOptions = "\nChoose an action: \n (1-9) - Enter an int

    ServerSocket serv = null;
    InputStream in = null;
    OutputStream out = null;
    Socket clientSocket = null;
    private final int id; // client id
    Game game; // current game

    private boolean inGame = false; // a game was started (you can decide if
    private String name; // player name

    private int currentState = 1; // I used something like this to keep track
    private static boolean grading = true; // if the grading board should be

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

4-2 Starter Code

SER 321

Protobufs

Option 2:
Command
Line

**Check out
the
recording
for the
discussion!**

The screenshot shows an IDE with the following components:

- Project View:** Shows the project structure for 'Assignment4.2given'. The 'src/main/java/server' directory is expanded, highlighting 'SockBaseServer'.
- Code Editor:** Displays the 'SockBaseServer.java' file. The code includes package declarations, imports, and a class definition for 'SockBaseServer' with static variables and a main method.
- Gradle View:** Shows the 'Tasks' list for 'Assignment4.2given'. The 'generateProto' task is highlighted.
- Terminal:** Shows the command 'gradle generateProto' being executed in the terminal window.

```
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 - to see 1
    static String gameOptions = "\nChoose an action: \n (1-9) - Enter an int

    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
```

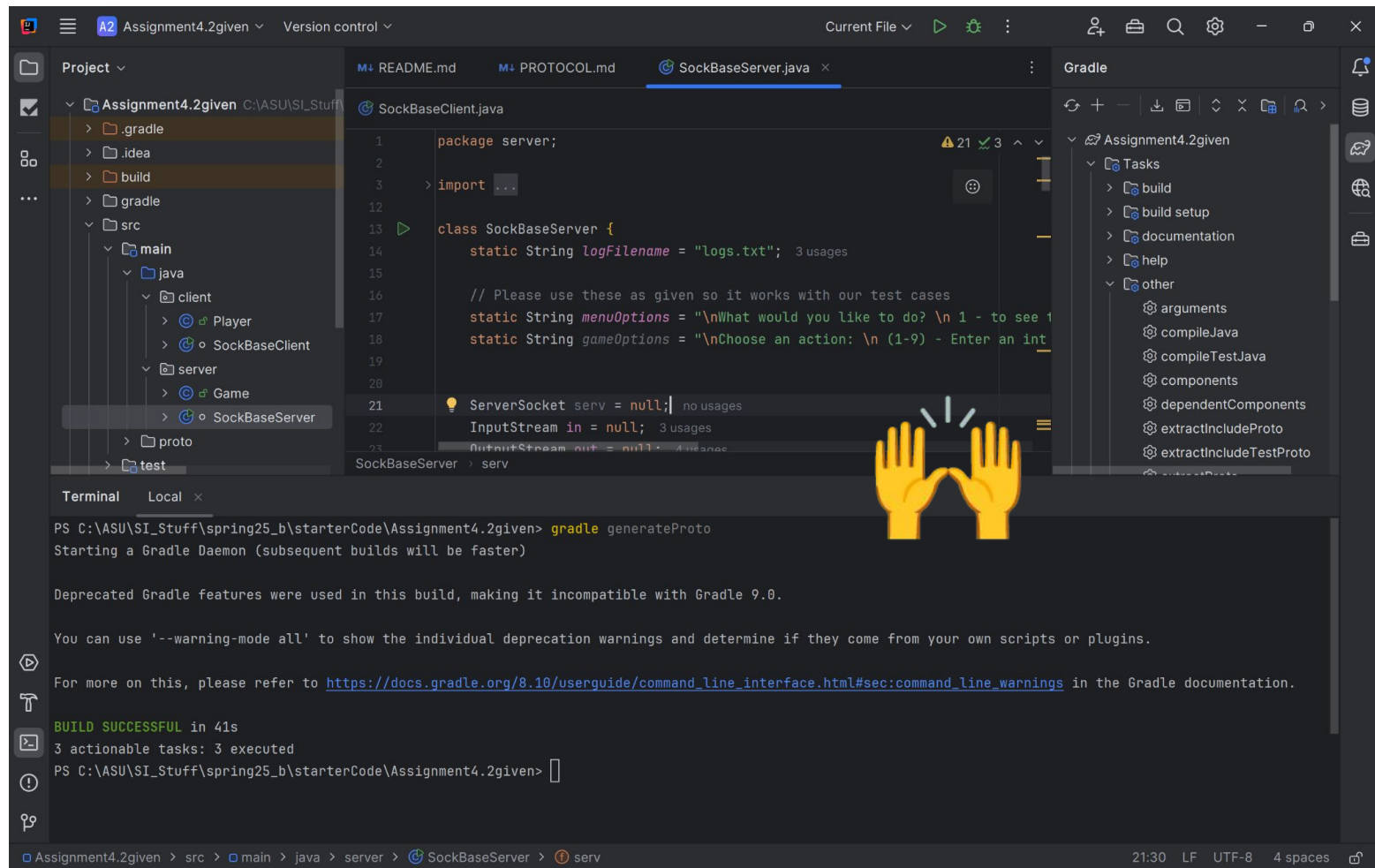
Terminal output: PS C:\ASU\SI_Stuff\spring25_b\starterCode\Assignment4.2given: **gradle generateProto**

4-2 Starter Code

SER 321

Protobufs

*Check out
the
recording
for the
discussion!*



Assignment4.2given

Project

- Assignment4.2given
- gradle
- idea
- build
- gradle
- src
 - main
 - java
 - client
 - Player
 - SockBaseClient
 - server
 - Game
 - SockBaseServer
 - proto
 - test

SocketBaseServer.java

```
1 package server;
2
3 import ...
4
12
13 class SockBaseServer {
14     static String logFilename = "logs.txt";
15
16     // Please use these as given so it works with our test cases
17     static String menuOptions = "\nWhat would you like to do? \n 1 - to see 1
18     static String gameOptions = "\nChoose an action: \n (1-9) - Enter an int
19
20
21     ServerSocket serv = null;
22     InputStream in = null;
23     OutputStream out = null;
```

Terminal

```
PS C:\ASU\SI_Stuff\spring25_b\starterCode\Assignment4.2given> gradle generateProto
Starting a Gradle Daemon (subsequent builds will be faster)

Deprecated Gradle features were used in this build, making it incompatible with Gradle 9.0.

You can use '--warning-mode all' to show the individual deprecation warnings and determine if they come from your own scripts or plugins.

For more on this, please refer to https://docs.gradle.org/8.10/userguide/command\_line\_interface.html#sec:command\_line\_warnings in the Gradle documentation.

BUILD SUCCESSFUL in 41s
3 actionable tasks: 3 executed
PS C:\ASU\SI_Stuff\spring25_b\starterCode\Assignment4.2given>
```

SER 321

Protobufs

Options for Message Creation

***Check out
the
recording
for the
discussion!***

```
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

Check out the recording for the discussion!

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;
        }
    }
}
```

SocketBaseClient - chooseMenu

```
Request.Builder req = Request.newBuilder();

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

SocketBaseClient - main

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

SER 321

Protobufs

Parsing Messages

GETTERS!

***Check out the
recording for
the
discussion!***

```
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());  
}
```

SER 321

System Layout

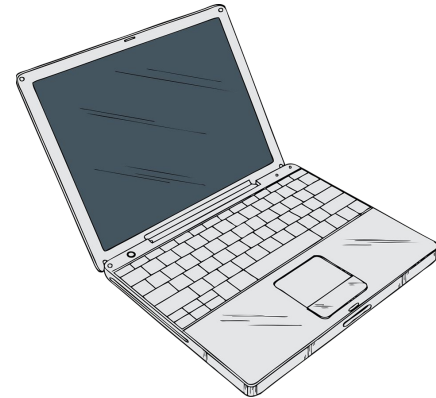
You have two systems...

***Check out
the
recording
for the
discussion!***

How can we test our server with multiple clients?



?

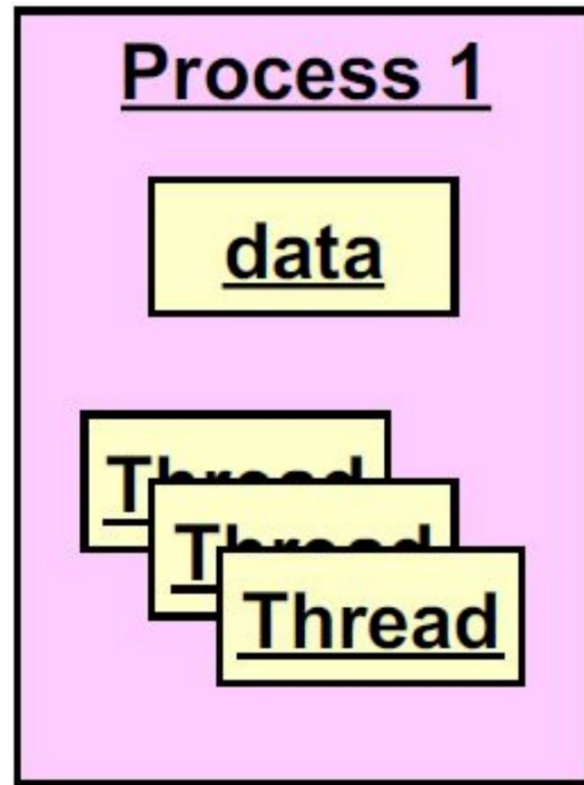
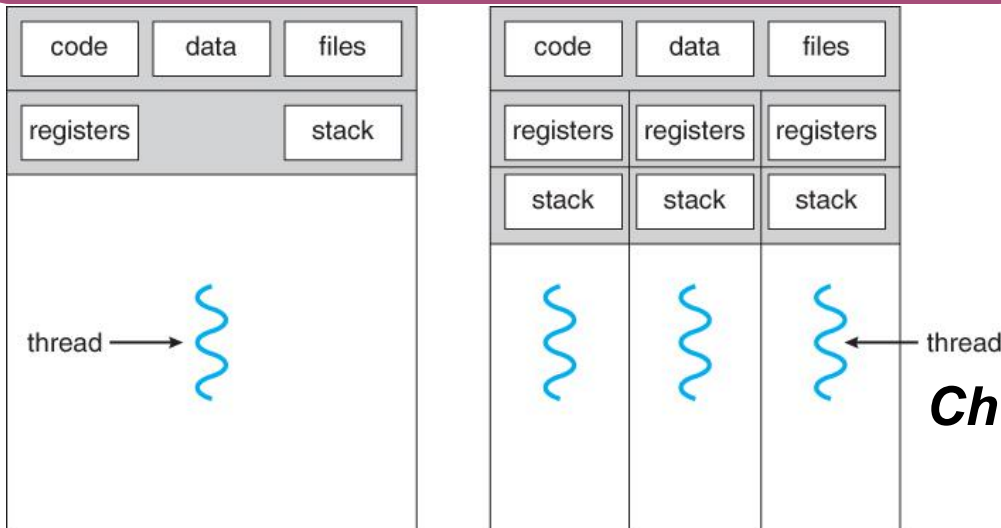


SER 321

Threads

What does that imply?

Remember that they exist *within* the parent process



Check out the recording for the discussion!

SER 321

Threading Pitfalls

Check out the recording for the discussion and 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

What's the difference?

Starvation

A thread never gains access to the resource it needs

Waiting to access the **CPU**

Ready to go; never gets a chance

vs.

***Check out
the
recording
for the
discussion!***

Deadlock

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

Waiting to access another **resource**

Not ready to go

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 and solution!

```
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 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

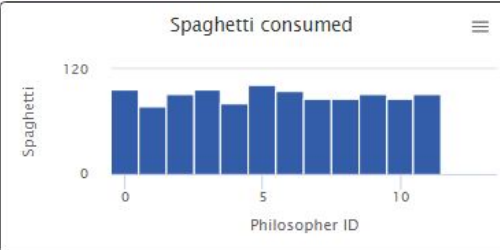
0.5

full-chance

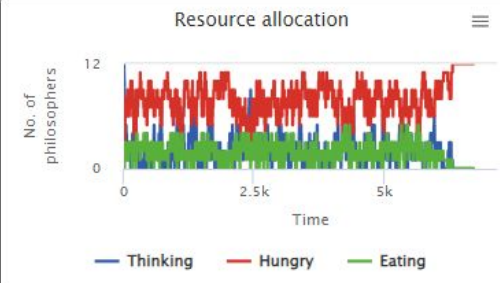
0.5

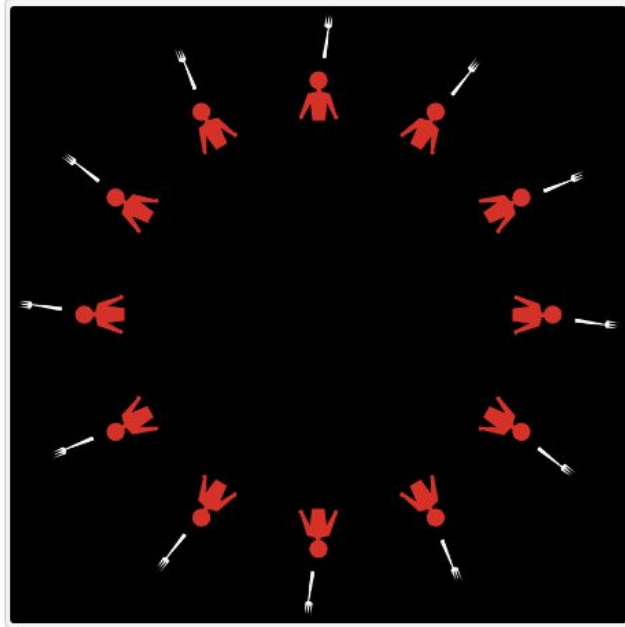
☐ cooperation?

Spaghetti consumed



Resource allocation





SER 321

Scratch Space

Upcoming Events

SI Sessions:

- Sunday, April 13th at 7:00 pm MST
- Tuesday, April 15th at 10:00 am MST
- Thursday, April 17th at 7:00 pm MST

Review Sessions:

- Sunday, April 27th at **6:00 pm MST - 2 hour Exam Review Session**
- Tuesday, April 29th, at 10:00 am MST - **Q&A Session**

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

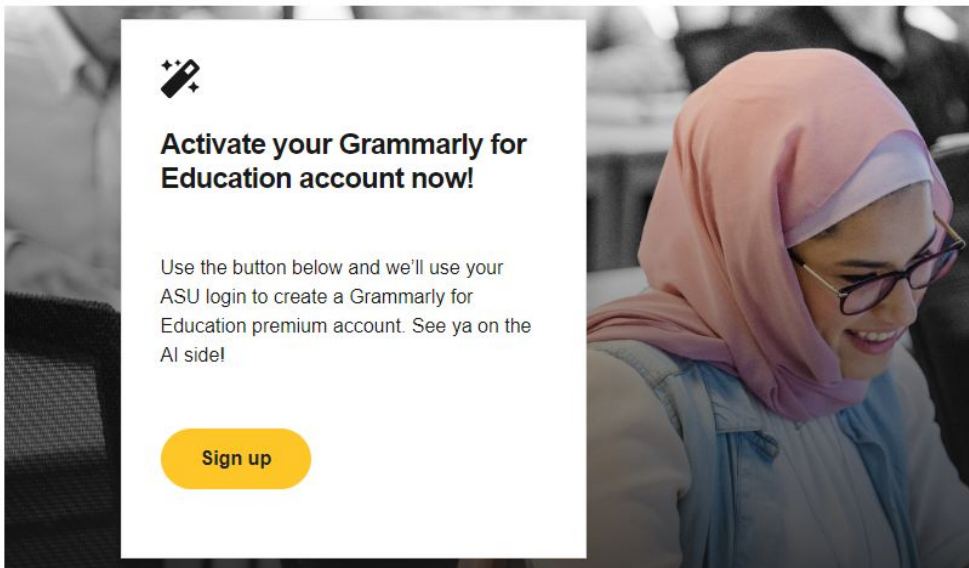
Computer Applications and Information Technology


[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!





Activate your Grammarly for Education account now!

Use the button below and we'll use your ASU login to create a Grammarly for Education premium account. See ya on the AI side!

[Sign up](#)



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](#)