This course supports the assessments for ABT1. The course covers 3 competencies and represents 3 competency units.

Introduction

Overview
Watch the following video for an introduction to this course:

- Introduction to Programming II (8 min.)

For this course, you will collaborate with a peer to develop a small application, conduct a review of a small object-oriented application that you developed, conduct a review of a design presentation that you created, and design and code an application that will perform create, read, update, and delete queries on a database.

As a competent programmer, your ability to design and develop algorithms, your proficient use of data structures, and your ability to use the Unified Modeling Language (UML) to communicate and develop object-oriented designs will help you design and develop applications to meet customer requirements. A solid understanding of object-oriented concepts will help you develop applications that are maintainable and extensible. Strong competence in software testing and troubleshooting will allow you to validate and verify your applications to ensure you are delivering a quality product that meets all requirements.

Java applications can run on any platform which is why it is well-suited for applications both big and small. Java is free, and it is everywhere. All of these reasons are why Java software developers are in high demand and can command an excellent salary. Being fluent in the Java programming language, knowledgeable of the Java API classes, and skilled in software development will qualify you to design and develop applications in a wide range of industries including education, energy, ecommerce, telecommunications, gaming, healthcare, and finance.

Watch the following video for an introduction to this course:

Competencies

This course provides guidance to help you demonstrate the following 3 competencies:

- **Competency 430.1.4: Modeling Systems Using Unified Modeling Language (UML)**
  The graduate develops and interprets Unified Modeling Language diagrams which model object-oriented designs.
- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.
- **Competency 430.1.6: Software Testing and Troubleshooting**
  The graduate applies software testing and troubleshooting strategies to determine

programming errors and recommend appropriate solutions.

Activities at a Glance Outline

The following outline will provide general guidelines for your work through the course and can be printed and used as a quick checklist for your progress:

- "Activities at a Glance"

Course Mentor Assistance
As you prepare to successfully demonstrate competency in this subject, remember that course mentors stand ready to help you reach your educational goals. As subject matter experts, mentors enjoy and take pride in helping students become reflective learners, problem solvers, and critical thinkers. Course mentors are excited to hear from you and eager to work with you.

Successful students report that working with a course mentor is the key to their success. Course mentors are able to share tips on approaches, tools, and skills that can help you apply the content you're studying. They also provide guidance in assessment preparation strategies and troubleshoot areas of deficiency. Even if things don't work out on your first try, course mentors act as a support system to guide you through the revision process. You should expect to work with course mentors for the duration of your coursework, so you are welcome to contact them as soon as you begin. Course mentors are fully committed to your success!

Preparing for Success

The information in this section is provided to detail the resources available for you to use as you complete this course.

Learning Resources

The learning resources listed in this section are required to complete the activities in this course. For many resources, WGU has provided automatic access through the course. However, you may need to manually enroll in or independently acquire other resources. Read the full instructions provided to ensure that you have access to all of your resources in a timely manner.

Automatically Enrolled Learning Resources

You can access the learning resources listed in this section by clicking on the links provided throughout the course. You may be prompted to log in to the WGU student portal to access the resources.

SkillSoft and Books 24x7
You will access SkillSoft items at the activity level within this course. For more information on accessing SkillSoft items, please see the "Accessing SkillSoft Learning Resources" page.

The following Books24x7 e-texts will be used in this course:

- Horstmann, C. *Big Java* (3rd ed.). Hoboken, NJ: Wiley. ISBN: 9780470105542.

Note: These e-texts are available to you as part of your program tuition and fees, but you may purchase hard copies at your own expense through a retailer of your choice. If you choose to do so, please use the ISBN listed to ensure that you receive the correct edition.

Note: The Big Java *e-text has a student companion site, which gives you access to rich tools and resources that are available for each chapter of this text. This includes source code for each chapter.*

Additional Preparations

NetBeans

You will be required to develop Java applications for this course. NetBeans is the recommended integrated development environment (IDE).

This IDE will provide you with an editor, debugger, and the Java Development Kit. If you use NetBeans, you will not have to download the Java SE Development Kit as a separate activity. You can download the SDK bundled with NetBeans (recommended).

Download the Java SE NetBeans IDE Download Bundle at the following website:

- NetBeans

MySQL

You will need MySQL Database Management Server installed. Please follow the installation instructions in the following document:

- "Installation Instructions For MySQL"

Create Notebook

It is suggested that you create a paper or digital notebook as you work through this course. Use organizers or dividers to separate your work. Suggested dividers include:

- Java syntax notes
- solutions to compiler errors encountered
- NetBeans commands
- Java API classes used

Study Group

Go to the Software Development message board and locate a study partner or two. This activity will provide you with peers with whom you can discuss the concepts you are learning and also build your professional network. This will make it easier for you to establish partnering for the performance assessment.

Getting Started With Unified Modeling Language (UML)

This subject will help you to understand how to use UML to model the requirements and the design of your system. UML provides a notation that allows software developers to design systems by creating graphical models. Both dynamic and static diagrams are provided with strict notation rules. Because the notation is unified, you will be able to communicate any design with any developer in any language in any region. Notation also provides a means for analyzing the requirements of a system prior to entering the design phase.

UML Modeling Tools

To help you create an object-oriented design, it can be helpful to model your system to help you identify the classes that you require.

Creating UML diagrams will help you determine the classes you need, the relationships between the classes, and how the objects in your application interact for a given scenario. UML use case diagrams will help you get a handle on how the application will be used. UML use case diagrams are an excellent way to help you document user requirements. Class diagrams and sequence diagrams are excellent for documenting an object-oriented design that will satisfy the given requirements.

This topic will introduce you to tools that will help you draw your diagrams using appropriate UML 2.0 notation.

Investigate UML Modeling Tools

Try a few tools to create use case, class, and sequence diagrams from the following document:

- "UML Tools"

After you have tried a few of these tools, please share with others the tool you will use to complete this course and post your recommendation to the Software Development message board.

If you have located an additional tool that you recommend, please feel free to share information on how to access this tool along with your feedback.

Using UML to Model Your Systems

This lesson will help you understand how to use UML to model the requirements of your system and the design of your system. Knowing all 13 standard diagram types will allow you to communicate and review your designs to ensure the application will meet specified requirements.

UML 2.0

This topic focuses on developing your knowledge of the notation you will use to create UML diagrams to model your systems. The activities will provide you with practice in developing

accurate UML diagrams. As you study each UML diagram, ask yourself the following questions:

- How is this diagram used to model the system?
- What is the correct notation used to create this diagram?

This topic addresses the following competencies:

- **Competency 430.1.4: Modeling Systems Using Unified Modeling Language (UML)** The graduate develops and interprets Unified Modeling Language diagrams which model object-oriented designs.

Research UML 2.0

Using the following website, read about UML 2.0:

- Unified Modeling Language

Appendix M in your text also provides a brief summary of UML.

As you read, consider the following questions:

- What are the different diagrams provided by UML 2.0?
- What is a static diagram?
- Which of the UML diagrams are considered static diagrams?
- What is a dynamic diagram?
- Which of the UML diagrams are considered dynamic diagrams?
- Which diagrams do you think are most used?
- Locate at least two links that show you the correct notation for each UML 2.0 diagram. Bookmark these links for your reference as you complete your UML diagrams in the upcoming tasks.

UML Use Case Diagrams

Conduct an Internet search to find examples of use case diagrams. Make notes on the correct UML 2.0 notation for use cases. You will have to draw these in your upcoming performance assessment. Find answers to the following questions:

- What does a use case diagram model?
- What is the correct notation for a use case diagram?
- What can be actors?
- How do you depict an actor?
- How many use case diagrams should you create?
- How do you represent the system under development in a use case?
- Can use case diagrams have inheritance? If so, why would you do this?

UML Class Diagrams

Conduct an Internet search to find examples of class diagrams. Make notes on the correct UML 2.0 notation for class diagrams. You will have to draw these in your upcoming performance

assessment. Find answers to the following questions:

- What does a comprehensive class diagram model?
- What is the correct notation for a class diagram?
- Where do you list the attributes?
- Where do you list the methods?
- How do you note access modifiers such as public and private?
- How do you note inheritance relationships?
- How do you note interface implementation?
- How do you note aggregation?
- How do you note multiplicity?

UML Sequence Diagrams

Conduct an Internet search to find examples of sequence diagrams. Make notes on the correct UML 2.0 notation for sequence diagrams. You will have to draw these in your upcoming performance assessment. Find answers to the following questions:

- What does a sequence diagram model?
- What is the correct notation for a sequence diagram?
- Is a sequence diagram a static or dynamic UML diagram?
- Where, in the sequence diagram, do you place the objects involved in the scenario?
- How do you represent an object's timeline in a sequence diagram?
- How do you represent messaging between objects in a sequence diagram?

Practice With UML Diagrams

Complete the following exercises:

1. Think of a real-world example of a system with users. Create a use case diagram that represents at least two uses of the system.
2. For your system, identify at least three objects that may interact in order for the system to provide its users with the functionality required. Create a comprehensive class diagram of this system. Include all attributes, methods, access modifiers, relationships, and any multiplicity.
3. For your system, create a least two sequence diagrams that model the interaction between the objects for at least two scenarios.

Object-Oriented Programming (Part I)

This section will provide you with an overview of creating object-oriented designs by using classes to represent state and behavior. You will design relationships between classes to utilize the principles of encapsulation, polymorphism, and inheritance.

Polymorphism

This topic covers polymorphism, which provides a means for creating flexible programs that are easy to maintain. Polymorphism is one of the most important and difficult concepts to apply. Even though you may understand the theory behind polymorphism, applying it requires another level of understanding. Understanding how to recognize opportunities to utilize polymorphism

and knowing how to implement polymorphism are some of the most important skills required to develop flexible and maintainable applications. Please devote an entire week to polymorphism and complete the suggested exercises. As you conduct your reading, ask yourself the following questions:

- How can specialized behavior be determined at runtime using polymorphism?
- How does polymorphism create flexible applications?
- Why should you try to program to an interface instead of programming to an implementation?
- What is the difference between overloading and overriding methods? How are these cases of polymorphism?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 9: Interfaces and Polymorphism

Read the following chapter in *Big Java*:

- chapter 9 ("Interfaces and Polymorphism")

Make sure you understand the principles behind polymorphism. Also, note how it is implemented in code. See if you can complete review exercises R9.3, R9.5, R9.7, and R9.8. See if you can complete programming exercises P9.3 and P9.9.

Try to answer the followingq uestions:

- What is an interface?
- Can multiple classes implement the same interface?
- What is polymorphism? How is it useful for code reuse?
- What is a comparable interface? Why would a class want to implement this interface?

Object-Oriented Programming (Part II)

This section will provide you with an overview of creating object-oriented designs by using classes to represent state and behavior. You will design relationships between classes to utilize the principles of encapsulation, polymorphism, and inheritance.

Inheritance

The principle of inheritance is used to create hierarchies of classes where a parent class is the generalization and the subclasses define the specializations. Inheritance is one of the most important and difficult concepts to apply. Even though you may understand the theory behind inheritance, applying it requires another level of understanding. Understanding how to recognize opportunities to utilize inheritance and knowing how to implement inheritance are some of the most important skills required to develop flexible and maintainable applications. The principle of

inheritance is widely applied throughout the Java API. Understanding this principle will help you understand the relationships between the classes in the Java API. Please devote an entire week to inheritance and complete the suggested exercises. As you complete this topic, ask yourself the following:

- How does inheritance promote code reuse?
- How does inheritance produce easy-to-maintain applications?
- How could you have used inheritance in previous applications that you have developed?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 10: Inheritance

Read the following chapter in *Big Java*:

- chapter 10 ("Inheritance")

As you read, pay close attention to the fields and methods that are inherited by the subclasses. Note how subclasses are implemented in Java. It is important that you understand the concept of inheritance as well as understand how to implement it in the Java programming language. Also, note the rules of inheritance in Java as they apply to multiple inheritance and what can or cannot be inherited from the parent class.

Complete review exercises R10.2-R10.8.

Then, complete exercises P10.1, P10.2, and P10.4.

As you complete this activity, consider the following questions:

- Can a subclass inherit from more than one class?
- Can a superclass be inherited by more than one class?
- What are the rules of inheriting instance fields and methods?
- Can you override methods from a superclass? If so, how would you do this?
- What do you use the "extends" keyword for?
- Why would you want to explicitly call the super constructor?
- What role does inheritance play with polymorphism?
- What is an abstract class? Why would you use one?
- What is an interface? How would you use one?
- What are the levels of access control?

Overriding the toString() Method

Understanding how to override the toString() method to print out your objects is very important.

Take time to practice implementing the toString() method by selecting any example of a class in the book. Create a test application that constructs objects of the class in which the toString() method was overridden. Use the printf() statement to print out your constructed objects' current state. An object's current state is represented by the current values of its attributes.

- Were you able to use the toString() method in a printf() statement to print out your object contents?

Task 1 Performance Task

Complete the following task in TaskStream:

- Intro to Programming II: Task 1

For details about this performance assessment, see the "Assessment Preparation" box in this course.

Task 2 Performance Task

Complete the following task in TaskStream:

- Intro to Programming II: Task 2

For details about this performance assessment, see the "Assessment Preparation" box in this course.

Multithreading

Multithreading is good to know because all modern processors are equipped with multiple cores on a chip. Using Multithreading enables programs to divide up the work that their program is doing to complete task faster across different cores on a chip.

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 20: Multithreading

Read the following chapter in *Big Java*:

- chapter 20 ("Multithreading")

As you read, pay close attention to how multiple threads are executed in Java. Note how threads are implemented in Java. It is important that you understand the concept of running threads and terminating threads. Also, note how to synchronize object accesses.

Complete review exercises R20.1, R20.2, R20.5, R20.7, and R20.8.

Then, complete exercises P20.1, P20.2, and P20.4.

Object-Oriented Design and the Software Development Process

This section will give you an overview of the software development process which includes eliciting requirements, design, development, and the very important phase of troubleshooting and testing. It is important that you understand the syntax of Java programming language and be familiar with as many of the API classes as possible. To be a great software developer, you need to know how to conduct each activity in the software development process and know how to complete the associated documentation. This knowledge will transform you from a programmer into a software engineer.

Phases of Software Development

The life cycle of software includes the phases of eliciting software requirements, design, development, and testing. Pay close attention to the activities that occur during each phase of the life cycle and the information captured in the corresponding documentation. As you complete the readings and tasks for this topic, ask yourself these questions:

- Do you know how and when to document requirements?
- Do you know how and when to document your design?
- Do you know how and when to document your test plan?
- Do you understand the purpose of each phase in the software life cycle?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 12: Object-Oriented Design

Read the following chapter in *Big Java*:

- chapter 12 ("Object-Oriented Design")

This chapter will orient you to the software development process and the activities for each phase. After reading the chapter, see if you can answer review exercises R12.1-R12.5.

Practice With Use Case Diagrams

While completing the activities in the "UML 2.0" topic earlier in this course you learned how to create use case diagrams. Use case diagrams help describe the requirements of an application to be developed and should be included in the Software Requirement Specification document.

Create at least three use case diagrams for the application described in exercise P12.6.

Practice With Class Diagrams

While completing the activities in the "UML 2.0" topic earlier in this course, you learned how to

create comprehensive class diagrams. Class diagrams help describe the design of the application and should be included in the Software Design Specification document.

For review exercise R12.13 (skip CRC cards), complete UML comprehensive class and use case diagrams.

Documentation

The Software Development Lifecycle produces many artifacts including documentation. As a developer in industry, you will be expected to generate and maintain documentation. The Software Requirements Specification (SRS) and the Software Design Specification (SDS) are two such documents. You are also likely to develop or use a Software Test Plan.

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Software Test Plan

Conduct Internet research and locate a Software Test Plan template. Bookmark this template as you will use this as you complete your performance assessment at the end of this course.

Software Design Patterns

Software design patterns are reusable designs that can be used to create solutions for your applications. In this subject section, you will research several software design patterns.

Software Design Patterns

In this topic, you will research several software design patterns and learn how to apply these to your software solutions. Developers that have good knowledge of the available design patterns will tend to lead design activities.

The information you gather during these activities will be used for the third task. Be sure you are following APA citation rules as you make notes during your research.

As you locate a software design pattern, reflect on the following questions:

- Is this software design pattern relevant for an object-oriented design?
- What types of applications can be designed using this pattern?
- What are the benefits of the pattern?
- What are the disadvantages of the pattern?
- Can you draw the architecture of this pattern?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Research Model View Controller Design Pattern

Conduct Internet research and answer the questions below. Capture the results in your notebook, as you will need this information for one of the tasks. Be sure to note the source of your information so you can properly cite it.

- What are the three components of the MVC pattern? What is the role of each component?
- What types of applications can be designed using the MVC pattern?

Research Software Design Patterns

Conduct Internet research and answer the following questions:

- Why are software design patterns beneficial?
- Identify and learn about at least three other software design patterns. For which types of applications would each pattern you researched be appropriate to use as a design?
- Who were the Gang of Four, and what was their influence on software design patterns?

Task 3 Performance Task

Complete the following task in TaskStream:

- Intro to Programming II: Task 3

For details about this performance assessment, see the "Assessment Preparation" box in this course.

Object-Oriented Design and Data Structures

This section will provide you with an overview of data structures and introduce you to collections. The Java API capitalized on the object-oriented principle of inheritance and polymorphism when designing collections.

Inheritance With Data Structures

The Java API capitalizes on inheritance and interfaces with the implementation of sets, lists, and maps. You will first learn about the characteristics of each of these data structures and then understand how Java created the collections hierarchy. Most Java applications are data intensive and likely will benefit from the incorporation of the data structures provided by the Java API. As you conduct your reading, ask yourself the following questions:

- What are the characteristics of the data structure?
- How could I use this data structure?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 15: An Introduction to Data Structures

Read the following chapter in *Big Java*:

- chapter 15 ("An Introduction to Data Structures")

Make sure you understand the characteristics of all data structures covered. Then try to answer the following questions:

- What are the characteristics of a list?
- What are the characteristics of a set?
- What are the characteristics of a map?
- What are the characteristics of a queue?
- What are the characteristics of a stack?

Chapter 16: Advanced Data Structures

Read the following chapter in *Big Java*:

- chapter 16 ("Advanced Data Structures")

Make sure you reference the Java API documentation for each collection covered in the chapter. Using the information in chapter 16 and referring to the Java API documentation of collections, can you answer the following questions?

- What implementations of a list does the Java API provide?
- What are at least two methods an ArrayList provides?
- What implementations of a set does the Java API provide?
- Can you sort a set? If so, how?
- What implementations of a map does the Java API provide?
- Why would you use a map?
- How is the principle of inheritance used to create the collection hierarchy?

Practice With Advanced Data Structures

To further your ability to use the collections, please complete the following programming exercises:

- P16.1
- P16.4

Object-Oriented Applications and Databases

Many of the object-oriented designed applications that you create will have a database backend. In this section, you will learn how to connect a Java application to a database and access the data. This section is one of the most challenging, so be sure to devote an entire week of dedicated study time to these activities. The skills developed in this section are critical to your successful completion of your upcoming performance assessment.

Connecting to a Database

The Java API provides classes that make connecting to a database straightforward. Many applications you will develop or maintain will be connected to one or more databases. You will learn how to connect to a database through completing these activities. Pay close attention to the SQL commands available to query, update, and create data. As you complete this topic, answer the following:

- Do you understand how to install a database?
- Do you understand how to connect to a database?
- Do you understand why applications would want to connect to a database?
- Do you understand the SQL syntax to query, add, delete, and update?
- Do you understand how to update, add, delete, and query data on a database from a Java application?

This topic addresses the following competencies:

- **Competency 430.1.5: Object-Oriented Concepts**
  The graduate applies object-oriented concepts, develops object-oriented designs, and uses object-oriented programming techniques.

Chapter 22: Relational Database

Read the following chapter in *Big Java*:

- chapter 22 ("Relational Database")

As you read, pay close attention to the SQL syntax.

Note: Skip the instructions (i.e., section 22.3) to install and connect to a database. *You were directed to install MySQL at the beginning of this course.*

Be sure to read section 22.4 ("Database Programming in Java") carefully. This explains how to use JDBC, how to execute SQL statements, and how to analyze query results. Understanding this material will help you with the next activity, Java, JDBC, and MySQL.

Java, JDBC, and MySQL

In this activity, you will first enhance an existing application before moving onto developing one of your own. Upon conclusion, you will know how to develop an application that creates, reads, updates, and destroys records in a database.

You are supplied with a sample Java application that manipulates a database. You are provided

a SQL script to create the database. To download the activity instructions, sample application, and SQL script, access the following ZIP file:

- database cos activities.zip

Testing Your Object-Oriented Solution

This section will give you an overview of the very important phase of the software life cycle of troubleshooting and testing. There are several phases and types of testing strategies to consider that will be covered.

Software Test Strategies

In this topic you will research several important testing strategies. This information will help you understand the appropriate time to apply a testing technique and the reasons to apply a testing technique. Thorough and effective testing results in applications that correctly meet all requirements, including functional and non-functional. As you read about each test activity, ask yourself the following questions:

- When would this testing strategy be used?
- Why is this type of testing beneficial?
- Who would conduct this type of testing?
- How does this testing strategy verify or validate the application?
- How would you plan to use this strategy?
- How would the results of this testing strategy be documented?

This topic addresses the following competencies:

- **Competency 430.1.6: Software Testing and Troubleshooting**
  The graduate applies software testing and troubleshooting strategies to determine programming errors and recommend appropriate solutions.

Research Black Box Testing

Read the following section in *Big Java*:

- section 5.5 of chapter 5 ("Decisions")

Conduct an Internet research on the topic of black box testing. As you conduct your research, be sure you can answer the following questions:

- What is black box testing?
- What are the benefits of this type of testing?
- When during the software life cycle is black box testing conducted?
- Who conducts black box testing?

Research White Box Testing

Read the following section in *Big Java*:

- section 3.6 of chapter 3 ("Implementing Classes")

Conduct an Internet research on the topic of white box testing. As you conduct your research, be sure you can answer the following questions:

- What is white box testing?
- What are the benefits of this type of testing?
- When during the software life cycle is white box testing conducted?
- Who conducts white box testing?

Research Integration Testing

Conduct Internet research on integration testing.

As you conduct your research, be sure you can answer the following questions:

- What is integration testing?
- What are the benefits of this type of testing?
- When during the software life cycle is integration testing conducted?
- Who conducts integration testing?

Research System Testing

Conduct Internet research on system testing.

As you conduct your research, be sure you can answer the following questions:

- What is system testing?
- What are the benefits of this type of testing?
- When during the software life cycle is system testing conducted?
- Who conducts system testing?

Research Regression Testing

Read the following section in *Big Java*:

- section 7.7 of chapter 7 ("Arrays and Array Lists")

Conduct Internet research on the topic of regression testing. As you conduct your research, be sure you can answer the following questions:

- What is regression testing?
- What are the benefits of this type of testing?
- When during the software life cycle is regression testing conducted?
- Who conducts regression testing?

Task 4 Performance Task

Complete the following task in TaskStream:

- Intro to Programming II: Task 4

For details about this performance assessment, see the "Assessment Preparation" box in this course.

Final Steps

Congratulations on completing the activities in this course! This course has prepared you to complete the assessments associated with this course. If you have not already been directed to complete the assessments, schedule and complete your assessments now.

The WGU Library

The WGU Library is available online to WGU students 24 hours a day.

For more information about using the WGU Library, view the following videos on The WGU Channel:

- WGU: Accessing the Library
- WGU Library: Finding Articles, Books, & E-Reserves

Center for Writing Excellence: The WGU Writing Center

If you need help with any part of the writing or revision process, contact the Center for Writing Excellence (CWE). Whatever your needs—writing anxiety, grammar, general college writing concerns, or even ESL language-related writing issues—the CWE is available to help you. The CWE offers personalized individual sessions and weekly group webinars. For an appointment, please e-mail writingcenter@wgu.edu.

Feedback

WGU values your input! If you have comments, concerns, or suggestions for improvement of this course, please submit your feedback using the following form:

- Course Feedback

ADA Requirements
Please review the University ADA Policy.