Preview: IT3348 : Intermediate Programming

Syllabus

Course Overview

In this course, you will demonstrate your programming knowledge and skills by using the C++ programming language to demonstrate common programming concepts, such as:

- Design patterns.
- Data abstraction.
- Inheritance.
- Polymorphism.
- Data structures.
- Algorithms.
- Recursion.

Course Competencies

(Read Only)

To successfully complete this course, you will be expected to:

1. Define business problems that can be solved using basic programming concepts and standards.
2. Analyze intermediate object-oriented programming.
3. Apply collaboration strategies in developing software components.
4. Design a program that applies intermediate programming concepts and constructs.
5. Communicate effectively.

Course Prerequisites

Learners who have received credit for IT3340 may not take IT3348 or IT-FP3348. Prerequisite(s): IT2240 or IT-FP2240; IT3225 or IT-FP3225.

Required

The materials listed below are required to complete the learning activities in this course.

Integrated Materials

Many of your required books are available via the VitalSource Bookshelf link in the courseroom, located in your Course Tools. Registered learners in a Resource Kit program can access these materials using the courseroom link on the Friday before the course start date. Some materials are available only in hard-copy format or by using an access code. For these materials, you will receive an email with further instructions for access. Visit the Course Materials page on Campus for more information.

Book

Prata, S. (2012). *C++ primer plus* (6th ed.). Upper Saddle River, NJ: Addison Wesley. ISBN: 9780321776402.

Suggested

The following materials are recommended to provide you with a better understanding of the topics in this course. These materials are not required to complete the course, but they are aligned to course activities and assessments and are highly recommended for your use.

External Resource

Please note that URLs change frequently. While the URLs were current when this course was designed, some may no longer be valid. If you cannot access a specific link, contact your instructor for an alternative URL. Permissions for the following links have been either granted or deemed appropriate for educational use at the time of course publication.

- Allain, A. (n.d.). Lesson 9: C strings. Retrieved from http://www.cprogramming.com/tutorial/lesson9.html

Optional

The following optional materials are offered to provide you with a better understanding of the topics in this course. These materials are not required to complete the course.

Library

The following optional readings may be available in the Capella University Library. To find specific readings by journal or book title, use Journal and Book Locator. Refer to the Journal and Book Locator library guide to learn how to use this tool. If the full text is not available, you may be able to request a copy through the Interlibrary Loan service.

- Skillsoft. (n.d.). C++ fundamentals [Tutorial].
- Skillsoft. (n.d.). Introduction to Eclipse [Tutorial].
- Skillsoft. (n.d.). Introduction to Eclipse Part 2 [Tutorial].
- Skillsoft. (n.d.). Object-oriented programming fundamentals [Tutorial].
- Skillsoft. (n.d.). Thinking defensively about functions, methods, and input [Tutorial].

External Resource

Please note that URLs change frequently. While the URLs were current when this course was designed, some may no longer be valid. If you cannot access a specific link, contact your instructor for an alternative URL. Permissions for the following links have been either granted or deemed appropriate for educational use at the time of course publication.

- Allain, A. (n.d.). Class design in C++. Retrieved from http://www.cprogramming.com/tutorial/class_design.html
- Allain, A. (n.d.). Lesson 10: C++ File I/O. Retrieved from http://www.cprogramming.com/tutorial/lesson10.html
- Allain, A. (n.d.). Lesson 11: Typecasting in C and C++. Retrieved from http://www.cprogramming.com/tutorial/lesson11.html
- Allain, A. (n.d.). Lesson 12: Introduction to classes in C++. Retrieved from http://www.cprogramming.com/tutorial/lesson12.html
- Allain, A. (n.d.). Lesson 19: Inheritance in C++. Retrieved from http://www.cprogramming.com/tutorial/lesson19.html
- Allain, A. (n.d.). Lesson 1: The basics of C++. Retrieved from http://www.cprogramming.com/tutorial/lesson1.html
- Allain, A. (n.d.). Lesson 20: C++ inheritance – syntax. Retrieved from http://www.cprogramming.com/tutorial/lesson20.html
- Allain, A. (n.d.). Lesson 2: If statements in C++. Retrieved from http://www.cprogramming.com/tutorial/lesson2.html
- Allain, A. (n.d.). Lesson 3: Loops. Retrieved from http://www.cprogramming.com/tutorial/lesson3.html
- Allain, A. (n.d.). Lesson 4: Functions. Retrieved from http://www.cprogramming.com/tutorial/lesson4.html
- Allain, A. (n.d.). Lesson 5: Switch case in C and C++. Retrieved from http://www.cprogramming.com/tutorial/lesson5.html
- Allain, A. (n.d.). Lesson 6: Pointers in C++. Retrieved from http://www.cprogramming.com/tutorial/lesson6.html
- Allain, A. (n.d.). Lesson 7: Structures in C++. Retrieved from http://www.cprogramming.com/tutorial/lesson7.html
- Allain, A. (n.d.). Lesson 8: Arrays in C and C++. Retrieved from http://www.cprogramming.com/tutorial/lesson8.html

- Allain, A. (n.d.). Understanding initialization lists in C++. Retrieved from http://www.cprogramming.com/tutorial/initialization-lists-c++.html

Projects

Project >> Team Collaboration Reflection

Project Overview

Programming in the real world requires quite a bit of collaboration. People and teams meet and work together on a regular basis to ensure the success of both small and large software development projects. Teams collaborate and lean on one another for support, a critical component of a software application's success within an organization.

Throughout the course, you will work as a team, collaboratting on the weekly exercises and assignments. Each week, you will find discussions forums specific to your team. Each person is required to contribute to these discussions.

- **Week 1:** You are asked to form into teams.
    - The instructor will assign you a team name of Eagles, Wolves, Bears, or Tigers. Each week you are expected to work with your team members, using the discussion corresponding with the name of your team. Your instructor may also set up a course group in the classroom that further supports your team collaboration: look in the course's My Groups area as well.
    - The weekly discussion is to help your team with debugging, developing, and optimizing code for your assignments each week. Share tips and resources to help one another write more efficient and effective code. You are expected to share ideas and statements, and support your team members throughout the course.
- **Weeks 2–5:** You and your team members will collaborate on completing the weekly assignments.
- **Week 5:** You will write a paper reflecting on your team experience, including the advantages and disadvantages of working on a software engineering team. In addition, you will be graded on your participation in the weekly team discussion.

Team Discussion Rules and Responsibilities

You are required to follow these rules when participating in the weekly team discussion:

- You will contribute substantively to your team discussion each and every week. Your participation must be **on time** for you to obtain credit for this portion of the assignment. There will be **no exceptions** to this rule.
- You (and everyone else) will contribute to the completion of each weekly exercise in a meaningful way.
- You are allowed to share snippets of code as you work on your assignments. The goal is to brainstorm and support one another in learning the concepts of the course.
- You are not allowed to share full code solutons, or offer soutions that hold no merit. It will be considered a violation of the course policies, for example, if code is shared without dialog or in a way that does not

promote the growth of each individual of the group. Your instructor will assess your posts peroidically and will determine the merit of a post.
- You are expected to read any Faculty Expectations for more guidelines regarding the team project.
- You are not allowed to join in on other team discussion areas. You may only work in your own team's discussion.

Your faculty may not monitor the team discussion every week. If you need faculty intervention, it is your responsibility to reach out and ask for a review of your team's discussion forum.

- Substantive participation in all team discussions is required along with the final paper.
- **Written communication:** Written communication is free of errors that detract from the overall message.
- **APA formatting:** Resources and citations are formatted according to APA (6th edition) style and formatting.
- **Font and font size:** Arial, 10 point.

Unit 1 >> Introduction to C++ and Object-Oriented Programming

Introduction

In this unit, you will write a simple program called Hello World in C++ using Code::Blocks, an integrated development environment (IDE) for C and C++.

A computer program is written in a language that is recognizable by humans. The language has a particular syntax, just like English or any other language. C++ uses a compiler to convert a computer program into machine code—a language written in ones and zeros—that a computer understands.

Object-oriented programming is an approach to writing software that models the real world in code. All models start with an object, which has properties and methods. Properties are the variables that the object requires. Methods are what the objects can do. When writing code, you create a class, which is a template of an object. A cookie cutter is an example of a class. The actual cookie that results from the cookie cutter is the object. So, the cookie cutter creates cookie objects. Cookies have sizes, kinds, and shape, which are the properties of the object. A method could be "eat."

Consider the following example of a C++ program:

```
1 // my first program in C++

2 #include <iostream>

3

4 int main()

5 {

6   std::cout << "Hello World!";

7 }
```

Line 1 is how you comment a C++ program. Line 2 starts with a hash sign (#), which is a directive to the C++ compiler to include a library in your program. Iostream is a class library that enables C++ programs to execute input and output. Lines 4 through 7 define the Hello World application. Line 6 is a statement. This statement says you are using the standard library and the cout method, which outputs something to the screen. Hello World will be output to the screen in this application.

Learning Activities

u01s1 - Studies

Readings

In this study's readings, you will be introduced to the C++ programming language and its history. You will also explore the key differences between procedural and object-oriented programming.

Use your *C++ Primer Plus* text to complete the following:

- Chapter 0, "Introduction," pages 1–8.
- Chapter 1, "Getting Started With C++," pages 9–25.
- Chapter 2, "Setting Out to C++," pages 27–62.
- Chapter 3, "Dealing With Data," pages 65–110.
- Chapter 4, "Compound Types," pages 115–191.

Optional – C++ Tutorials

The following optional resources provide helpful information about the topics in this unit. These were written by Alex Allain and appear on the Web site, Cprogramming.com.

- Lesson 1: The Basics of C++.
- Lesson 2: If Statements in C++.
- Lesson 3: Loops.
- Lesson 5: Switch Case in C and C++.

Optional Skillsoft Resources

- Skillsoft. (n.d.). Introduction to Eclipse [Tutorial].
- Skillsoft. (n.d.). Introduction to Eclipse Part 2 [Tutorial].

u01s2 - Software Preparation and Technology Access

In this course, you will be using software and technology that is needed to complete designated activities and assignments. There is no additional cost for this software and technology. Some software packages will be made available to you at no additional cost through Capella's subscription with Microsoft, while other software packages are available for free download through open-source licensing.

Capella University requires learners to meet certain minimum computer requirements. Please note that some software required for a course may exceed these minimum requirements. Check the requirements for the software you may need to download and install to make sure it will work on your device. Most software will require a Windows PC. If you use a Mac, refer to Installing a Virtual Windows Environment.

The software and technologies below are strongly recommended to support you in completing the course objectives. If you have access to other tools that you believe may still meet course requirements or if you have any difficulties accessing this resource or completing the related assignments, please contact your course faculty member to discuss potential alternatives.

If you use assistive technology or any alternative communication methods to access course content, please contact DisabilityServices@Capella.edu with any access-related questions or to request accommodations.

For this course, follow the instructions provided through the links below to download and install software or register for an account, as required.

Open-Source Software

- Code Blocks: Go to Downloading and Installing Code::Blocks With C/C++.

If you encounter any difficulties in the download and installation process, post a detailed question in the Ask Your Instructor section of the course. Your instructor should be able to help you or point you in the right direction for the answers you need.

u01a1 - Variables, Expressions, and Data Types

Assignment Context

By successfully completing this assignment, you will demonstrate your proficiency in the following course competencies and objectives:

- Competency 2: Analyze intermediate object-oriented programming.
    - Define the variables in an application.
    - Define the expressions in an application.
    - Explain the data types in an application.
- Competency 5: Communicate effectively.
    - Communicate effectively in a business environment.

Assignment Overview

For this assignment, you will write your first C++ application, called calcMileage. The goal of this program is to determine miles-per-gallon based on the number of miles driven and the number of gallons of gas you put in the tank. You will use the Code::Blocks software to write this application. Access this software using the instructions found in the Software Preparation and Technology Access study in this unit.

Assignment Instructions

Part 1: Write, Run, and Test the Application

Use the code in the Unit 1, Part 1 Assignment Instructions document linked in the Resources to complete this part of your assignment. Be sure to test the application with a variety of values before you proceed to Part 2. The program produces accurate results with some numeric values but not with others. Note when the results are accurate and when they are not.

Part 2: Improve the Application

Based on the material in the study, modify the code to fix the accuracy problem. Be sure to comment and format the code.

Part 3: Describe the Application

After running your application and observing the results, write a 1–2 page paper in which you complete the following:

- Define the variables in an application.
- Define the expressions in an application.
- Explain the data types in an application.
- Discuss the cause of the inaccuracy in the original code and the approach used to fix the problem.

Ensure that your writing is professional and consistent with the standards and conventions of the IT community.

Submit both the .cpp source code file and the Word document with the paper.

Submission Requirements

Refer to the Variables, Expressions, and Data Types Assessment Scoring Guide to ensure that you meet the grading criteria for this assessment.

Your completed assignment should include the following:

- Demonstrate the ability to apply math in a program.
- Define the variables in an application.
- Define the expressions in an application.
- Explain the data types in an application.
- Communicate effectively in a business environment.

When complete, compress your C++ program into a ZIP archive and submit your archive in the assignment area. Be sure to name your archive using the following format IT3348_u01a1_FirstName_LastName.zip

u01d1 - Eagles Team Collaboration

Welcome, Eagles, to your first efforts of collaboration!

In this discussion, take a minute to get to know one another and begin tackling your first set of required exercises.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 1 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

For this discussion:

- Introduce yourself to your group.
- Describe your past programming experience (if any).
- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u01d2 - Wolves Team Discussion

Welcome, Wolves, to your first efforts of collaboration!

In this discussion, take a minute to get to know one another and begin tackling your first set of required exercises.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 1 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

For this discussion:

- Introduce yourself to your group.
- Describe your past programming experience (if any).
- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u01d3 - Bears Team Discussion

Welcome, Bears, to your first efforts of collaboration!

In this discussion, take a minute to get to know one another and begin tackling your first set of required exercises.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 1 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

For this discussion:

- Introduce yourself to your group.
- Describe your past programming experience (if any).
- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u01d4 - Tigers Team Discussion

Welcome, Tigers, to your first efforts of collaboration!

In this discussion, take a minute to get to know one another and begin tackling your first set of required exercises.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 1 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

For this discussion:

- Introduce yourself to your group.
- Describe your past programming experience (if any).
- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.

- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 2 >> Branching, Logic, and Functions

Introduction

In this unit, you will examine the flow of simple programs.

You make decisions every day, such as what to wear and what you are going to eat. You can write a computer program to make decisions, too. Consider the following example.

You have been asked by the local department of motor vehicles to automate their drivers license testing process. The test consists of 20 questions. To pass the test, 13 of 20 questions must be answered correctly. So, let us write a decision statement in C.

Here is the template:

```
if ( statement is TRUE )
Execute this line of code
```

Here is a code fragment:

```
correct=13;
if (correct>13) cout << "You have passed the exam!";
if (correct<13) cout << "You have failed the exam!";
```

In this example, the person passed, so the cout statement "You have passed the exam!" runs. If you were to change the statement correct=13 to correct=5, then the cout statement "You have failed the exam!" runs.

In the following example, the previous code fragment is rewritten using an "else" statement, which is better than using "if" statements.

```
correct=13;
if (correct>13) {
 cout << "You have passed the exam!";
}
Else {
 cout << "You have failed the exam!";
}
```

The "if" statement helps control program flow. The second way to control program flow is by using loops, such as the "for" or "while" loops. For example, you likely repeat certain activities every day. On work days you may wake up, shower, dress, eat breakfast, and drive to work. Using the "while" statement looks like this:

```
while (you are not dead) {
wake_up();
shower();
dress();
eat_breakfast();
drive_to_work(); …
}
```

Let us play a simple guessing game in which we will try to guess the age of a person 20 times. Once the age is guessed, the program exits and prints out a message. The program flow would look like this:

```
int correct;
int age;
correct=0;
while (correct<20) {
 cout << "Please guess an age that I am thinking about" ; /* Asks for age */
 cin >> age;
if (age==10)
 correct=20;
}
if (correct==20)
{
 cout << "You have failed!";
}
else
{
 cout << "You have passed!";
}
```

These examples provide an overview of program flow, also known as control structures. The two basic control structures are "if" statements and loops.

If you remember from algebra, f(x) = x + 1 is called a function—f(1) results in 2; f(2) results in 3. You will explore functions in C++ in this unit, as well as control structures.

Learning Activities

u02s1 - Studies

Readings

In this study's readings, you will learn how to control program flow through the use of logic and branching statements. Additionally, you will learn how to organize and reuse code through function constructs.

Use your *C++ Primer Plus* text to complete the following:

- Chapter 5, "Loops and Relational Expressions," pages 195–249.
- Chapter 6, "Branching Statements and Logical Operators," pages 253–298.
- Chapter 7, "Functions: C++'s Programming Modules," pages 305–372.

Optional – C++ Tutorials

The following optional resources provide helpful information about the topics in this unit. These were written by Alex Allain and appear on the Web site, Cprogramming.com.

- Lesson 4: Functions.
- Lesson 8: Arrays in C and C++.
- Lesson 9: C Strings.

Optional Skillsoft Resources

- Skillsoft. (n.d.). Thinking defensively about functions, methods, and input [Tutorial].

u02a1 - Branching, Logic, and Functions

Assessment Context

By successfully completing this assignment, you will demonstrate your proficiency in the following course competencies and objectives:

- Competency 1: Define business problems that can be solved using basic programming concepts and standards.
  - Define the requirements for an application.
  - Describe the stakeholders needed to further define an application.
  - Describe the ethical issues relevant to an application.
- Competency 2: Analyze intermediate object-oriented programming.
  - Analyze the behavior of basic, procedural-based programming constructs.
  - Explain how an application works.

- Competency 4: Design a program that applies intermediate programming concepts and constructs.
    - Write a program that applies basic data structures.
- Competency 5: Communicate effectively.
    - Communicate effectively in a business environment.

Assignment Overview

For this assignment, you will write a C++ application called calcPay for a financial company. The goal of this program is to determine gross pay for a 4-week pay period based on an hourly rate and the number of hours worked in each week of the pay period. Anyone working over 40 hours a given week earns time-and-a-half. You will use the Code::Blocks software to write this application. Access this software using the instructions found in Unit 1.

Assignment Instructions

Part 1: Write the Application

Write a C++ console application that asks the user to enter the hourly wage and number of hours worked for each week in a 4-week pay period. Create a custom function with appropriate input parameters and return type that calculates the total gross pay based on the values entered by the user. Use suitable loops for the user input and the processing of data entered by the user. The program should print out the gross pay for the pay period for the employee.

Part 2: Describe the Application

Write a 3–5-page paper in which you complete the following:

- Define and document the requirements for the calcPay application.
- Describe the stakeholders needed to help further define the application.
- Describe a collaboration plan for working with stakeholders.
- Analyze the behavior of basic, procedural-based programming constructs.
- Define the functions you have used in your application.
- Define the IF statements you have used in your application.
    - Explain how the IF statements work.
- Define the loops you used in your program.
    - Explain how the loops work.
- Explain how your application works and any issues you encountered with the application.
- Include a screen shot of your application, using the Print Screen function, to show that your application works.

Submit both the source code file(s) and the Word document with the paper.

Submission Requirements

Refer to the Branching, Logic, and Functions Assignment Scoring Guide to ensure that you meet the grading criteria for this assignment.

Your completed assignment should include the following:

- Write a program that applies basic data structures.
- Define the requirements for an application.
- Describe the stakeholders needed to further define an application.
- Describe a collaboration plan for working with stakeholders.
- Analyze the behavior of basic, procedural-based programming constructs.
- Demonstrate the ability to apply math in a program.
- Explain how an application works.
- Communicate effectively in a business environment.

When complete, compress your C++ program into a ZIP archive and submit your archive in the assignment area. Be sure to name your archive using the following format IT3348_u02a1_FirstName_LastName.zip

u02d1 - Eagles Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 2 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.

- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u02d2 - Wolves Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 2 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u02d3 - Bears Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 2 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u02d4 - Tigers Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 2 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 3 >> Memory, Objects, and Classes

Introduction

In this unit, you will examine memory, objects, and classes. In Unit 1, you saw how classes are templates of objects. An example of a template is a cookie cutter, and an example of an object is a cookie made from the cookie cutter. In C++, a class is referenced such as:

```
class <class_name> {
 public || protected || private:
  member1;
 public || protected || private:
member2;
…
public || protected || private:
 memberN;
};
```

A member is a variable declaration or a function declaration. A public declaration preceding a member says that it is seen by everyone. A private declaration says that it is seen only by the class itself and no one else. A protected declaration says that members are accessible by other members of the same class or derived classes.

```
#include <string>;
   class Person {
 private:
  string name;
  int age;
 public:
  void set_values(string,int);
  Person(string,int);
}
```

The Person class has two private properties: (1) name and (2) age, and a public function called set_values. The Person class also has a constructor called Person, which initializes an object when it is created.

Consider the following sample application:

```
// example: class constructor
#include <iostream>
using namespace std;

class Rectangle {
  int width, height;
 public:
  Rectangle (int,int);
  int area () {return (width*height);}
};
Rectangle::Rectangle (int a, int b) {
 width = a;
 height = b;
}

int main () {
 Rectangle rect (3,4);
 Rectangle rectb (5,6);
 cout << "rect area: " << rect.area() << endl;
 cout << "rectb area: " << rectb.area() << endl;
 return 0;
}
```

This sample application defines a Rectangle class and defines two rectangles: (1) rect and (2) rectb, and prints the area of the two rectangles. The class declaration defines two properties: (1) width and (2) height, which are needed to calculate the area of rectangle. There is a constructor that initializes the values of the width and height when an object is created. So, when Rectangle rect(3,4) runs, it sets the width at 3 and the height at 4. Notice that after you define a rectangle you reference its functions and properties via rect.area(); the name of the object, the property, or a function. These properties and functions must be public to be accessible.

Run this sample application in Code::Blocks. It will print the areas of two rectangles.

Learning Activities

u03s1 - Studies

Readings

In this study's readings, you will learn about the fundamental basic concepts of objects and classes. You will also learn how to manage program memory and organizing your programs through the use of namespace constructs.

Use your *C++ Primer Plus* text to complete the following:

- Chapter 8, "Adventures in Functions," pages 379–443.
- Chapter 9, "Memory Models and Namespaces," pages 447–498.
- Chapter 10, "Objects and Classes," pages 505–558.
- Chapter 11, "Working With Classes," pages 563–622.

Optional – C++ Tutorials

The following optional resources provide helpful information about the topics in this unit. These were written by Alex Allain and appear on the Web site, Cprogramming.com.

- Lesson 6: Pointers in C++.
- Lesson 7: Structures in C++.
- Lesson 11: Typecasting in C and C++.
- Lesson 12: Introduction to Classes in C++.

Optional Skillsoft Resources

- Skillsoft. (n.d.). C++ fundamentals [Tutorial].

u03a1 - Memory, Objects, and Classes

Assessment Context

By successfully completing this assignment, you will demonstrate your proficiency in the following course competencies and objectives:

- Competency 1: Define business problems that can be solved using basic programming concepts and standards.
    - Define the requirements for an application.
    - Describe the stakeholders needed to further define an application.
- Competency 2: Analyze intermediate object-oriented programming.
    - Analyze the behavior of object-oriented programming constructs.
    - Explain how an application works.
- Competency 3: Apply collaboration strategies in developing software components.
    - Describe a collaboration plan for working with stakeholders.
- Competency 4: Design a program that applies intermediate programming concepts and constructs.
    - Write a program that applies object-oriented programming constructs.
- Competency 5: Communicate effectively.
    - Communicate effectively in a business environment.

Assignment Overview

For this assignment, you will write a C++ application called Area. The goal of this program is to determine the area of a circle and a square. You will use the Code::Blocks software to write this application. Access this software using the instructions found in the study in Unit 1.

Assignment Instructions

Part 1: Research

Research how to calculate the area of a circle and square.

Part 2: Write the Application

Follow the instructions in the Unit 3, Part 2 Assignment Instructions document, linked in the Resources, to complete this part of your assignment. You should organize your code into header files (Circle.h, Square.h) and implementation files (Circle.cpp, Square.cpp, main.cpp). Be sure to save a screen shot of your working application.

Part 3: Describe the Application

Write a 3–5-page paper in which you complete the following:

- Define and document the requirements for the Area application.
- Describe the stakeholders needed to help further define the application.
- Describe a collaboration plan for working with stakeholders.
- Analyze the behavior of classes and objects constructs.
- Define the properties, methods, and constructors you have used in your application
- Explain how your application works and any issues you encountered with the application.

- Include a screen shot of your application, using the Print Screen function, to show that your application works.

Submission Requirements

Refer to the Memory, Objects, and Classes Assignment Scoring Guide to ensure that you meet the grading criteria for this assignment.

Your completed assignment should include the following:

- Write a program that applies object-oriented programming constructs.
- Define the requirements for an application.
- Describe the stakeholders needed to further define an application.
- Describe a collaboration plan for working with stakeholders.
- Analyze the behavior of object-oriented programming constructs.
- Explain how an application works.
- Communicate effectively in a business environment.

When complete, compress your C++ program into a ZIP archive and submit your archive in the assignment area. Be sure to name your archive using the following format IT3348_u03a1_FirstName_LastName.zip

u03d1 - Eagles Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 3 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u03d2 - Wolves Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 3 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u03d3 - Bears Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 3 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u03d4 - Tigers Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 3 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 4 >> Object Inheritance and Dynamic Memory Allocation

Introduction

In this unit, you will examine object inheritance and dynamic memory allocation. Take the example of the genes you inherited from your parents. That concept of inheritance is similar in C++. A child class can inherit from a parent class. The child inherits any public and protected members of that class. The syntax for inheritance is as follows:

```
class Aclass {
 modifier: member;
 modifier: function();
}

class Bclass : public Aclass {
…
}
```

To declare a class, you use the following format:

```
bobj = new BClass();
delete bobj;
```

This dynamically creates both a pointer to a class and memory locations to store your object when it is declared. The delete statement frees up the memory assigned to bobj. You must do this to ensure that you keep your programs lean.

Inheritance is an important concept in C++ along with allocating and deallocating memory. If you define a pointer, be sure to clean up after you are done with it.

Learning Activities

u04s1 - Studies

Readings

In this study's readings, you will learn about the fundamental basic concpets of single and multiple class inheritance as well as dynamic memory allocation.

Use your *C++ Primer Plus* text to complete the following:

- Chapter 12, "Classes and Dynamic Memory Allocation," pages 627–700.
- Chapter 13, "Class Inheritance," pages 707–779.

Optional – C++ Tutorials

The following optional resources provide helpful information about the topics in this unit. These were written by Alex Allain and appear on the Web site, Cprogramming.com.

- Lesson 19: Inheritance in C++.
- Lesson 20: C++ Inheritance – Syntax.
- Class Design in C++.
- Understanding Initialization Lists in C++.

u04a1 - Inheritance and Dynamic Memory

Assignment Context

By successfully completing this assignment, you will demonstrate your proficiency in the following course competencies and objectives:

- Competency 1: Define business problems that can be solved using basic programming concepts and standards.
    - Define the requirements for an application.
    - Describe the stakeholders needed to further define an application.
- Competency 2: Analyze intermediate object-oriented programming.
    - Analyze the behavior of object-oriented programming constructs.
    - Explain how an application works.
- Competency 3: Apply collaboration strategies in developing software components.
    - Describe a collaboration plan for working with stakeholders.
- Competency 4: Design a program that applies intermediate programming concepts and constructs.
    - Write a program that applies object-oriented programming constructs.
- Competency 5: Communicate effectively.
    - Communicate effectively in a business environment.

Assignment Overview

For this assignment, you will write a new version of the area calculation program from Unit 3 that makes use of inheritance in C++. Add a new Shape base class to the area calculation program that includes data members common to all shapes (such as a shape ID, a shape type, and a unit of measure). You will use the Code::Blocks software to write this application. Access this software using the instructions found in Unit 1.

Assignment Instructions

Part 1: Write the Application

Write a new version of the area calculation program from Unit 3 that makes use of inheritance in C++.

- Add a new Shape base class to the area calculation program that includes data members common to all shapes (such as a shape ID, a shape type, and a unit of measure).
- The Shape base class should also include a virtual getArea() member function.
- Revise the Circle and Square classes so that they inherit from the Shape base class.
- Add a third shape to the program that also inherits from the Shape class.
- The finished program should ask the user to enter the relevant information for each shape and then print the area and other information for each shape.

You will use the Code::Blocks software to write this application. Be sure to organize the code correctly into header (.h) and implementation (.cpp) files. Your code should include meaningful comments and be correctly formatted.

Part 2: Describe the Application

Write a 3–5-page paper in which you complete the following:

- Define and document the requirements for the calcMortgage application.
- Describe the stakeholders needed to help further define the application.
- Describe a collaboration plan for working with stakeholders.
- Explain your object model and inheritance.
- Explain how your application works and any issues you encountered with the application.
- Include a screen shot of your application, using the Print Screen function, to show that your application works.

Submit all source code files and the Word document with the paper.

Submission Requirements

Refer to the Inheritance and Dynamic Memory Assignment Scoring Guide to ensure that you meet the grading criteria for this assignment.

Your completed assignment should include the following:

- Write a program that applies object-oriented programming constructs.
- Define the requirements for an application.
- Describe the stakeholders needed to further define an application.
- Describe the collaboration plan for working with stakeholders.
- Analyze the behavior of object-oriented programming constructs.
- Explain how an application works.
- Communicate effectively in a business environment.

When complete, compress your C++ program into a ZIP archive and submit your archive in the assignment area. Be sure to name your archive using the following format IT3348_u04a1_FirstName_LastName.zip

u04d1 - Eagles Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 4 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u04d2 - Wolves Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 4 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.

- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u04d3 - Bears Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 4 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u04d4 - Tigers Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 4 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

Unit 5 >> Code Reuse, Exceptions, and Input/Output

Introduction

In this unit, you will explore code reuse, exceptions, and input/output (I/O). Up to this point, you have been using I/O with cout and cin objects in your applications. You will continue to use those objects in your programs. The primary purpose of creating classes and using inheritance is code reuse. C++ provides extensive libraries that you can use and inherit from. You are reusing code when you do this. When designing an application, it is important that you reuse code whenever possible.

You must be able to handle C++ exceptions in your code. The compiler handles compile time issues, but what happens when you run into a runtime error? C++ provides an elegant mechanism to handle exceptions in your code. Consider the following example:

```
// exceptions
#include <iostream>
using namespace std;
 int main () {
    try
 {
  throw 20;
 }
 catch (int e)
 {
  cout << "An exception occurred. Exception Nr. " << e << '\n';
 }
 return 0;
}
```

This example shows a throw statement and a try … catch statement, which throws an exception 20; and because it is inside the try … catch block, the exception is caught and an error is printed on the screen.

Learning Activities

u05s1 - Studies

Readings

In this study's readings, you will learn how to manage and handle exceptional program events in your programs through the use of exception constructs.

Use your *C++ Primer Plus* text to complete the following:

- Chapter 13, "Reusing Code in C++," pages 785–869.
- Chapter 14, "Friends, Exceptions, and More," pages 877–947.

Optional – C++ Tutorials

The following optional resource provides helpful information about the topics in this unit. It was written by Alex Allain and appears on the Web site, Cprogramming.com.

- Lesson 10: C++ File I/O.

Optional Skillsoft Resources

- Skillsoft. (n.d.). Object-oriented programming fundamentals [Tutorial].

u05a1 - Exception Handling

Assignment Context

By successfully completing this assignment, you will demonstrate your proficiency in the following course competencies and objectives:

- Competency 2: Analyze intermediate object-oriented programming.
    - Analyze the behavior of exception handlers.
    - Explain how an application works.
- Competency 4: Design a program that applies intermediate programming concepts and constructs.
    - Modify a program to execute exception handling.
- Competency 5: Communicate effectively.
    - Communicate effectively in a business environment.

Assignment Overview

For this assignment, you will modify the area calculation application you wrote in Unit 4 to handle exceptions. You will use the Code::Blocks software to write this application. Access this software using the instructions found in Unit 1.

Assignment Instructions

Part 1: Research

Research how to catch different kinds of exceptions in C++. Pay particular attention to using custom exception classes that inherit from std::exception. You may wish to use the information in this unit's studies as a starting point for your research.

Part 2: Modify the Area Calculation Application

Create 2 custom exception classes that inherit from std::exception to handle invalid dimensions for shapes (such as 0 and negative numbers). Revise the code in your program's classes and man() to use these custom exception classes.

Part 3: Describe the Application

Write a 2–3 page paper in which you complete the following:

- Explain how you handled the exceptions in the code for your application.
- Explain how your application works and any issues you encountered with the application.
- Include a screen shot of your application, using the Print Screen function, to show that your application works.

Submission Requirements

Refer to the Exception Handling Assignment Scoring Guide to ensure that you meet the grading criteria for this assignment.

Your completed assignment should include the following:

- Modify a program to execute exception handling.
- Analyze the behavior of exception handlers.
- Explain how an application works.
- Communicate effectively in a business environment.

When complete, compress your C++ program into a ZIP archive and submit your archive in the assignment area. Be sure to name your archive using the following format IT3348_u05a1_FirstName_LastName.zip

u05a2 - Team Collaboration Reflection

Please reflect on your experience working as a group. Explain what you learned about collaboration and teamwork with software development.

Include the following in your discussion:

- Describe how the collaboration discussions aided you in completing the weekly exercises.
- Define the overall advantages of software development colllaboration.
- Describe challenges that you encountered working as a team.
- Describe the overall disadvantages of collaborating on a software development project.
- Identify the strategies and techniques that you and your team applied to ensure success. How do you manage to include every member of your team?

u05d1 - Eagles Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 5 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u05d2 - Wolves Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 5 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u05d3 - Bears Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 5 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide

u05d4 - Tigers Team Discussion

Welcome back! Please continue your collaboration and programming activities.

In these discussions, you will collaborate while completing your weekly assignments. At the end of the course you will be asked to reflect on your experience working in teams. You will be graded on this team assignment at the end of the course, but your participation in these discussions will be calculated into the overall score.

Use this discussion to work with your team members working through the Unit 5 assignment. Be sure to adhere to the Team Discussion Rules and Responsibilities as laid out in the course project description. Help one another with debugging, developing, and optimizing code for your assignment. Share tips and resources to help one another write more efficient and effective code. Share ideas, statements, and support one another throughout the course.

Do not forget to check the My Groups area of your course page to see if your instructor has posted any other resources for your team and remember, keep your participation here. Do *not* participate in other teams' discussions.

Discussion Guidelines

- Post questions relevant to this week's exercises and tasks.
- Share insights and resources related to this week's exercises and tasks.
- Help your team troubleshoot installation and programming issues.
- Collaborate to complete weekly required exercises.

Course Resources

Undergraduate Discussion Participation Scoring Guide