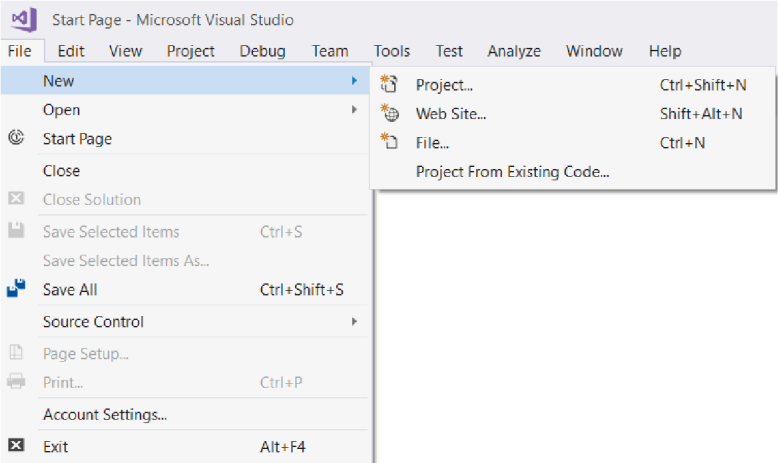
|  |  |
| --- | --- |
| **SUBJECT:** | CS442AL – PE CURRENT PROGRAMMING APP |
| **ACTIVITY:** | ACTIVITY # 7: EXERCISES 1, 2, 3, 4 & 5 – WRITING CONSOLE APPLICATIONS DEMONSTRATING DO, WHILE & FOR STATEMENTS INCLUDING THE CLAUSES.    NOTE: Prior to develop below applications, make sure that the following directives are included in your C# Class app:    using System;  using System.Collections.Generic; using System.Linq; using System.Text;  using System.Threading.Tasks; using static System.Console; using static System.Convert;    REMINDER: PLEASE PUT ALL ACCOMPLISHED PROGRAM CODES (AS USUAL) OF EXERCISES INCLUDING THIS FILE WITH AN ESSAY ANSWERS, INTO OUR ONEDRIVE SHARED FOLDER [ACTIVITY7](https://adamsoneduph-my.sharepoint.com/:f:/g/personal/renato_baisa_adamson_edu_ph/Et6I6R53LUhLuOghI1tX6RcBhvL-91zVphHxQz8GiXIGYw?e=edac9a)    Please use the usual file name convention, which is ACT07\_FML.zip |

# EXERCISE 01: Creating a Simple Console Application Using DO loop statement: Ch07Ex01\Program.cs

**Objectives:**

* **Create a new project Console Application.**
* **Learn how to use the ‘do’ loop and its effect to program flow.**
* **Run and display the output as shown below.**

1. Create a new console application project by selecting File ➪ New ➪ Project, as shown in Figure 1.



1. Ensure that the Visual C# node is selected in the left pane of the window that appears, and choose the Console Application (.NET Framework) project type in the middle pane. Change the Location text box to C:\CS442AL\Activity07\ (this directory is created automatically if it doesn’t already exist). Write in the Name text box as Ch07Ex01.
2. Click the OK button.
3. Once the project is initialized, add the following lines of code to the file displayed in the main window. **Your code statements must be identical below.**

double balance, interestRate, targetBalance; WriteLine("What is your current balance?");

balance = ToDouble(ReadLine());

WriteLine("What is your current annual interest rate (in %)?");

interestRate = 1 + ToDouble(ReadLine()) / 100.0; WriteLine("What balance would you like to have?"); targetBalance = ToDouble(ReadLine()); int totalYears = 0; do {

balance \*= interestRate;

++totalYears;

}

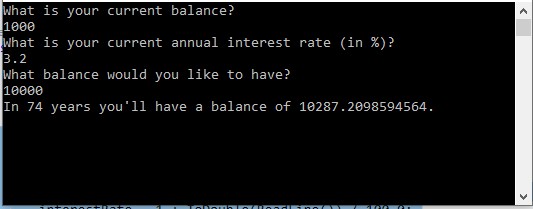
while (balance < targetBalance);

WriteLine($"In {totalYears} year{(totalYears == 1 ? "" : "s")} " +

$"you'll have a balance of {balance}.");

ReadKey();

1. Select the Debug ➪ Start Debugging menu item. After a few moments you should see the window shown in Figure 2.



**Figure 2**

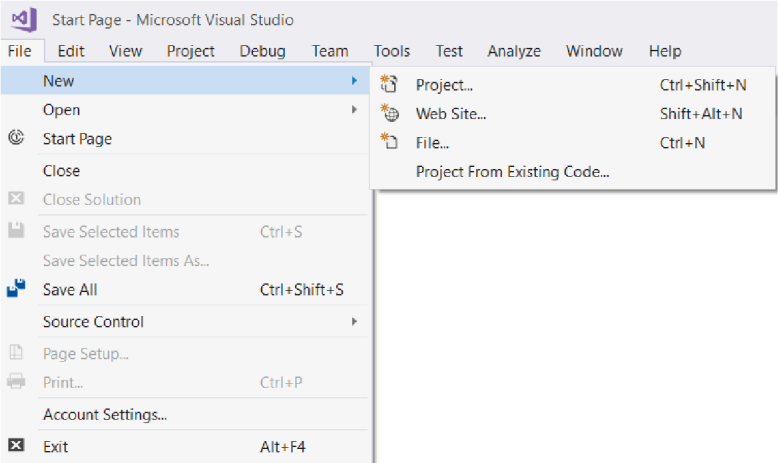
1. Press any key to exit the application (you might need to click on the console window to focus on it first).

# EXERCISE 02: Creating a Simple Console Application Using WHILE loop statement: Ch07Ex02\Program.cs

**Objectives:**

* **Create a new project Console Application.**
* **Learn how to use the “while” loop and its effect to program flow.**
* **Run and display the output as shown below.**

1. Create a new console application project by selecting File ➪ New ➪ Project, as shown in Figure 3.



1. Ensure that the Visual C# node is selected in the left pane of the window that appears, and choose the Console Application (.NET Framework) project type in the middle pane. Change the Location text box to C:\CS442AL\Activity07\ (this directory is created automatically if it doesn’t already exist). Write in the Name text box as Ch07Ex02.
2. Click the OK button.
3. Once the project is initialized, add the lines of code to the file displayed in the main window. **Your code statements must be identical below.**

double balance, interestRate, targetBalance; WriteLine("What is your current balance?");

balance = ToDouble(ReadLine());

WriteLine("What is your current annual interest rate (in %)?");

interestRate = 1 + ToDouble(ReadLine()) / 100.0; WriteLine("What balance would you like to have?"); targetBalance = ToDouble(ReadLine());

int totalYears = 0;

while (balance < targetBalance)

{

balance \*= interestRate;

++totalYears;

}

WriteLine($"In {totalYears} year{(totalYears == 1 ? "" : "s")} " +

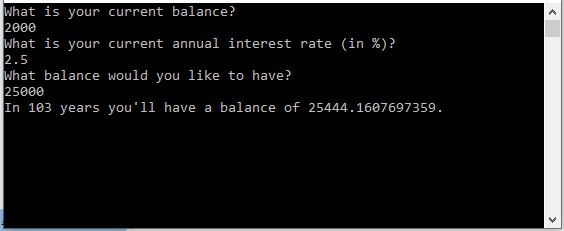
$"you'll have a balance of {balance}.");

if (totalYears == 0)

WriteLine(

"To be honest, you really didn't need to use this calculator."); ReadKey();

1. Select the Debug ➪ Start Debugging menu item. After a few moments you should see the window shown in Figure 4.



**Figure 4**

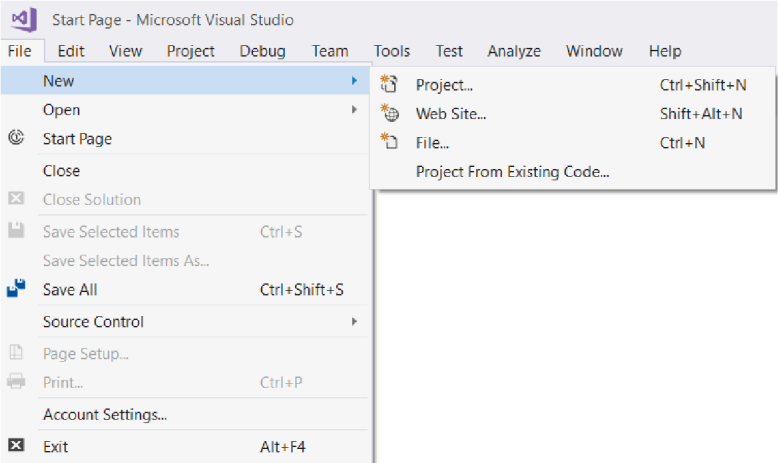
1. Press any key to exit the application (you might need to click on the console window to focus on it first).

# EXERCISE 03: Creating a Simple Console Application Using FOR loop statement: Ch07Ex03\Program.cs

**Objectives:**

* **Create a new project Console Application.**
* **Learn how to use the “for” loop and its effect to program flow.**
* **Run and display the output as shown below.**

1. Create a new console application project by selecting File ➪ New ➪ Project, as shown in Figure 5.



1. Ensure that the Visual C# node is selected in the left pane of the window that appears, and choose the Console Application (.NET Framework) project type in the middle pane. Change the Location text box to C:\CS442AL\Activity07\ (this directory is created automatically if it doesn’t already exist). Write in the Name text box as Ch07Ex03.
2. Click the OK button.
3. Once the project is initialized, add the lines of code to the file displayed in the main window. **Your code statements must be identical below.**

double balance, interestRate, targetBalance; WriteLine("What is your current balance?");

balance = ToDouble(ReadLine());

WriteLine("What is your current annual interest rate (in %)?");

interestRate = 1 + ToDouble(ReadLine()) / 100.0; WriteLine("What balance would you like to have?");

targetBalance = ToDouble(ReadLine());

int totalYears = 0, i;

for (i = 1; balance < targetBalance; i++)

{ balance \*= interestRate;

++totalYears;

}

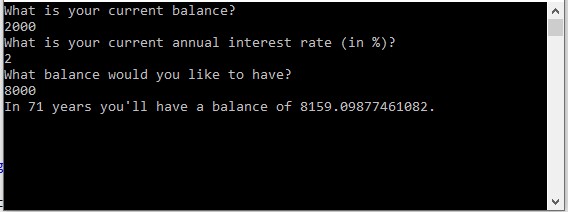
WriteLine($"In {totalYears} year{(totalYears == 1 ? "" : "s")} " +

$"you'll have a balance of {balance}."); if (totalYears == 0)

WriteLine(

"To be honest, you really didn't need to use this calculator."); ReadKey();

1. Select the Debug ➪ Start Debugging menu item. After a few moments you should see the window shown in Figure 6.



**Figure 6**

1. Press any key to exit the application (you might need to click on the console window to focus on it first).
2. How will you differentiate the looping statements do, while and for in terms of effect in program flow of execution?

While loop and for loop are two distinct looping constructs used in the provided code to accomplish the similar task of estimating the time needed to reach a target balance with a given interest rate. While and for loops both repeatedly run a block of code until a given condition—balance < targetBalance, for example—becomes false. Up until it reaches or surpasses the target balance, the balance is updated within the loop based on the interest rate. The primary difference is in the loops' syntax and organization, where the for loop provides a more condensed method of combining loop initialization, condition, and iteration. Despite these variations, all loops facilitate iterative execution until the desired condition is met, which adds to the program's flow.

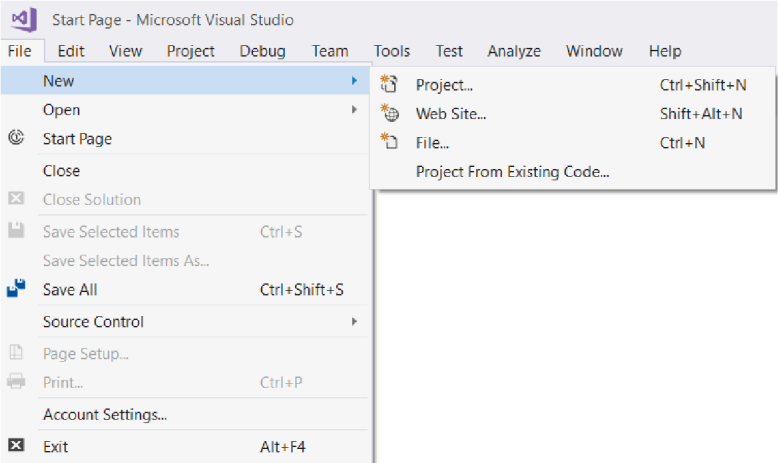
**EXERCISE 04: Creating a Console Application Using CONTINUE clause in FOR loop:**

# Ch07Ex04\Program.cs

**Objectives:**

* **Create a new project Console Application.**
* **Practice to use the “continue” clause in “for” loop statement.**
* **Determine the significance of using “continue” in the loop.**
* **Run and display the output as shown below.**

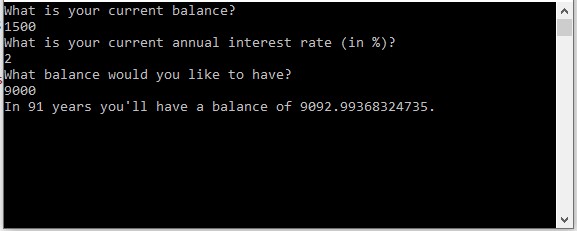
1. Create a new console application project by selecting File ➪ New ➪ Project, as shown in Figure 7.



1. Ensure that the Visual C# node is selected in the left pane of the window that appears, and choose the Console Application (.NET Framework) project type in the middle pane. Change the Location text box to C:\CS442AL\Activity07\ (this directory is created automatically if it doesn’t already exist). Write in the Name text box as Ch07Ex04.
2. Click the OK button.
3. Once the project is initialized, add the lines of code to the file that using the “continue” clause.
4. Use any of the above codes but amend the loop statement with continue clause. The condition to continue is as follows:

if (totalYears == 10) continue;

1. Select the Debug ➪ Start Debugging menu item. After a few moments you should see the window shown in Figure 8.



**Figure 8**

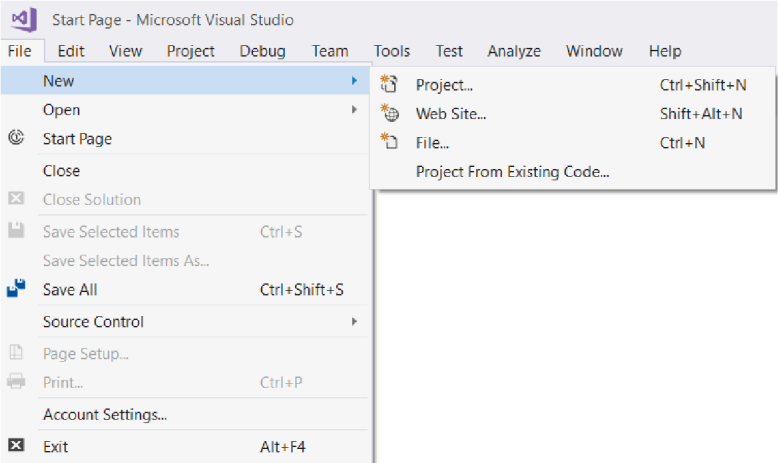
1. Press any key to exit the application (you might need to click on the console window to focus on it first).

# EXERCISE 05: Creating a Console Application Using BREAK clause in FOR loop: Ch07Ex05\Program.cs

**Objectives:**

* **Create a new project Console Application.**
* **Practice to use the “break” clause in “for” loop statement.**
* **Determine the significance of using “break” in the loop.**
* **Run and display the output as shown below.**

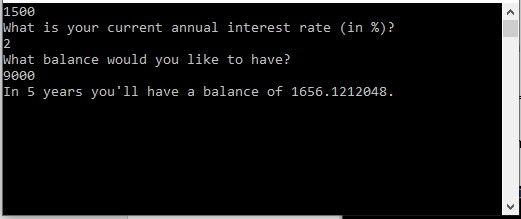
1. Create a new console application project by selecting File ➪ New ➪ Project, as shown in Figure 9.



1. Ensure that the Visual C# node is selected in the left pane of the window that appears, and choose the Console Application (.NET Framework) project type in the middle pane. Change the Location text box to C:\CS442AL\Activity07\ (this directory is created automatically if it doesn’t already exist). Write in the Name text box as Ch07Ex05.
2. Click the OK button.
3. Once the project is initialized, add the lines of code to the file that using the “continue” clause.
4. Use any of the above codes but amend the loop statement with continue clause. The condition to break the cycle is as follows:

if (totalYears == 5) break;

1. Select the Debug ➪ Start Debugging menu item. After a few moments you should see the window shown in Figure 10.



**Figure 10**

1. Press any key to exit the application (you might need to click on the console window to focus on it first).
2. How will you differentiate the break and continue clauses in for loop statements?

When a particular condition (totalYears == 5) is met, the for loop is broken early using the break statement, guaranteeing that the loop ends after a predefined number of iterations. On the other hand, when another condition (totalYears == 10) is satisfied, the continue statement is used to skip the remaining code inside the loop for that iteration, enabling the loop to move on to the next iteration without running the following code block. This distinction enables controlled flow within the loop: continue skips the current iteration and moves on to the next, while break completely exits the loop under specific circumstances.

Page