**Description: OC_Masterbrand_CMYK.eps**

in partnership with

****

ICT40515 Certificate IV in Programming

[Let’s](https://opencolleges.my.salesforce.com/a0QE0000009b11Z) get programming with C#

**SP5/Module 8 Assessment**

ICTPRG406 Apply introductory object-oriented language skills

ICTPRG415 Apply skills in object-oriented design

ICTPRG413 Use a library or pre-existing components

Assessment: 33129/02

*© Open Colleges Pty Ltd, 2016*

*All rights reserved. No part of the material protected by this copyright may be reproduced or utilised in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from the copyright owner.*

*All terms mentioned in this text that are known to be trademarks or service marks have been appropriately capitalised. Use of a term in this text should not be regarded as affecting the validity of any trademark or service mark.*

**Assessment task 33129/02**

## Introduction

This assessment will test your skills and knowledge gained through completing the learning and activities in **Module 8: Let’s get programming with C#.**

This is a two-part assessment.

In the first part of the assessment, you will answer a number of questions which will test your knowledge on C#, Object Oriented Design and Reusability.

In the second part of the assessment, you will analyse a requirements brief and develop an application in Visual Studio and C#.

Please use the checklist at the end of this document to ensure you have provided all the necessary pieces of evidence.

**Note – you must successfully complete ALL assessments (33129/01 and 33129/02) and have been assessed as competent in the previous assessments (33128/01 and 33128/02) from Module 6, Analyse Software Requirements and Module 7, Technical documents to achieve competency in:**

***ICTPRG406 Apply introductory object-oriented language skills***

***ICTPRG415 Apply skills in object-oriented design***

***ICTPRG413 Use a library or pre-existing components***

***Note to Assessor:***

***The attached are my notes and therefore my thought process while working through the assignment.***

***As a note there were no accessible files on GitHub and the Person.cs file / zip wasn’t relevant in the Additional Resources section. As such I started from scratch and created the Person.cs file.***

***Lance Flatman***

**CASE STUDY**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Acme Insurance Company**  https://s-media-cache-ak0.pinimg.com/564x/11/c2/92/11c29263df7ad88a1047a0bb9e5dab02.jpg  **Source:** [**Pinterest**](https://au.pinterest.com/pin/411657222161967801/)  **Outline**  The Acme Insurance Company sells a range of insurance products – from life insurance to car insurance to home and contents insurance. The company has approximately 50 employees, which includes both full time workers and contractors. The company’s headquarters is situated in Coyote Canyon where the local government is constantly changing the income tax rates. As such, the Acme Insurance Company has hired you to create a tax calculator program which can work out the tax to be paid on the income earned by their employees.  Your program must address the complexity of the different tax rate levied upon those employed full time compared to those who are employed on a contract basis. Those employed full time are taxed on a progressive scale (that is, different brackets of income are taxed at different rates, with higher income earners paying a greater percentage of tax). The current tax rates for weekly income are:     |  |  | | --- | --- | | **Income Bracket** | **Tax Rate** | | 500 | 5% | | 1,000 | 10% | | 1,500 | 15% | | 2,000 | 20% | | 3,000 | 30% |   For example, a full time employee who earns $1,700 (40 hours multiplied by a rate of $42.50) a week will pay $190 in tax. That is, they pay $25 for the first $500 earned at 5%, $50 for the next $500 earned at 10%, $75 for the next $500 at 15% and finally, $40 for the remaining $200 at 20%.  These income bracket and tax rates are contained in the *Rates.txt* file, which can be accessed [here](https://app.box.com/s/v2l2cd1mwhemijbigv88nyfk592rjei0). The first line contains the income brackets, with each value separated by a comma. The second line contains the tax rates, with each value separated by a comma.  Contractors pay a flat rate of 20%. As such, a contractor who earns $1,700 a week will pay $340 in tax.  You are required to capture the following information before calculating the tax payable amount:  Employee ID  First Name  Surname  Gender  Department  Email  Hourly Rate  Full time employees are paid according to a 40-hour payroll week. Conversely, for contractors you are required to capture the hours worked in the payroll week. A value must be entered into all fields.  *Employee ID* must begin with either a “E” or a “C” to signify whether the employee is full time or a contractor. This must be followed by a four-digit number (that is greater than 1000). For example, E1234 or C5678. The gender is either male or female. The department is to be selected from a drop down list (combo box). The list values are:  Accounts  Customer Service  IT  Administration  Once the Employee’s details have been entered, you are to calculate the tax payable. The employee’s details and the tax payable are to be written to a PDF file for which the user is able to select the location and filename to which it can be saved.  You are to provide a class diagram of the *Employee* and *Contractor* objects, and any refinements (such as generalisations), to the class diagram.  Appropriate comments are to be placed throughout your code.  You are to use the debugging features to ensure the code operates correctly. Following this, develop a test plan and test your application. |

**Tasks 1–5**

**Create Tax Calculator application**

Read the case study above thoroughly. Highlight, make notes, or extract the important information about the requirements put forth by the Acme Insurance Company. In the following tasks, you will need to use this information to create the requested application and associated documentation.

|  |
| --- |
| **Task 1: Develop the class diagram**  First, create a class diagram for the *Person*, *Employee* and *Contractor* class. The *Person* class will contain the *First name*, *Surname* and *Gender* attributes. Following this, apply generalisation, specialisation and inheritance principals to the three classes to refine the class diagram. Ensure you include methods and relationships.  **Submit:**   * **Person, Employee and Contractor class diagram.** * **Refined Person, Employee and Class diagram.** |

|  |
| --- |
| **Task 2: Evaluate Reuse components**  The requirement which requires addressing is to create a PDF file that reports the employee’s details and the tax payable. You are to research and evaluate at least three existing components that you can incorporate into Visual Studio to create a PDF file. One of these components must be iTextSharp.  Download the [reuse component evaluation](https://app.box.com/s/da95dvzwgi2xccpu1hzcpfb9lap886uh) template where you are to document your research and evaluation findings. Rename the reuse component evaluation template document to Evaluation\_yourName\_studentNumber.docx. (For example, Evaluation\_JohnSmith\_17756433.docx).  Your research and evaluation should include aspects such as technical implications and functionality, costs and licensing conditions. Finally, make a recommendation as to which component should be used.  **Submit:**   * **Evaluation document**   \* Irrespective of your recommendation, you will be using iTextSharp in the following tasks. |

|  |
| --- |
| **Task 3: Setup project**  **Task 3.1**  Create a Windows form project named *TaxCalculator*.  **Task 3.2**  From Github, download the Person.cs file from the OpenColleges repository and add it to the project. Modify the *Person* class to firstly add the *Employee* class, which is to inherit from the *Person* class. Then, modify the *Person* class to add the *Contractor* class, which is to inherit from the *Employee* class. Ensure to add any associated create methods.  **Task 3.3**  Install iTextSharp via the NuGet package manager. iTextSharp will be used to create the PDF output file. |

|  |
| --- |
| **Task 4: Create code**  **Task 4.1**  Create a form that will capture the combined details of the *Employee* and *Contractor* class. Note that *Hours Worked* is to only appear if an *Employee ID* is entered which begins with a *C* (use the *Leave* event to determine if the *Hours Worked* field is to be displayed).  Create four buttons to: create the *Employee*, calculate the employee’s tax payable amount, create the *Contractor* and calculate the contractor’s tax payable amount.  Add a multiline text box to display the entered values when the *Employee* and *Contractor* are created.  Ensure all controls are named according to naming standards.  **Task 4.2**  Write the code to create the *Employee* and *Contractor* class. Once created, display the entered details into the multiline textbox. Ensure appropriate edit checks are made before creating *Employee* or *Contractor*. Add appropriate comments.  **Task 4.3**  Write the code to calculate the tax payable for both the *Employee* and the *Contractor*. Income brackets and tax rates are to be read from the [Rates.txt](https://app.box.com/s/v2l2cd1mwhemijbigv88nyfk592rjei0) file and placed in arrays to calculate the *Employee* tax. To find out how to do this, research how to use the methods ReadLines() and Split(). Both of these call a method to create a PDF file that reports the Employee’s ID, combined name, department, salary and tax payable. Research how to create a PDF file using iTextSharp. The user is to select a location in which to save the PDF file. The *Save As* dialog should list a default filename and extension. Add appropriate comments.  **Submit:**   * **Project Source files** * **Two PDF files, one for the tax payable for an Employee and one for the tax payable for the Contractor.** |

|  |
| --- |
| **Task 5: Testing your application.**  You are required to test your program throughout the development cycle using de-bugging tools. In recording this, you are to complete the [testing template](https://app.box.com/s/0hbaqedvx45zvre579cp5qjyvua7e6qi) provided. Rename the testing template document to TestCase\_yourName\_studentNumber.docx. (For example, TestCase\_JohnSmith\_17756433.docx).  For the testing document, include the name of the application, your name, the date of when the test was carried out and a brief description of what you are going to test in *Test Specifications*. For each test, assign a sequential number to the ID column, what you are testing, how you are going to perform the test, the steps in performing the test, what you will be inputting, the expected and actual result and if the test passed or failed.  Create at least four test cases using the testing template to test the outcomes of the following:   * Add Employee * Calculate Employee tax payable * Add Contractor * Calculate Contractor tax payable.   You can do multiple tests for each of the above.  **Submit:**   * **Testing document with results** |

**Next steps for you:**

You have now reached the end of this assessment.

Ensure that you have completed all of the above tasks. Use the checklist to double check that you have submitted everything required for this assessment.

Zip all your files and upload to OpenSpace. Name your file 33129-02\_yourName\_studentNumber.zip. (For example, 33129-02\_JohnSmith\_17756433.zip).

Your trainer will provide you with feedback for this assessment – ensure that you read this feedback carefully.

**Checklist of items to submit as evidence for this assessment:**

|  |  |  |
| --- | --- | --- |
| **PART A** | | |
| TASK 1 | Employee and Contractor class diagram | 🞏 |
| TASK 1 | Refined Employee and Contractor class diagram | 🞏 |
| TASK 2 | Reuse component evaluation document | 🞏 |
| TASK 4 | Project source code | 🞏 |
| TASK 4 | Two PDF files, one for the tax payable for an Employee and one for the tax payable for the Contractor. | 🞏 |
| TASK 5 | Testing document | 🞏 |
| **Name each assessment file appropriately and submit for marking.** | | |