

6G671111 CW1 - 50 %

You are required to create a online application for a mortgage broker and to document your testing of it. In the sequel the testing document is referred to as test-harness.pdf

0.1 client-side application

Using the web technologies and data validation techniques that you meet in the lectures this term you are required to create a web application that will determine how much an online building society will lend to a user and the range of products that are available. All mortgage offers are shown in table 1.

0.1.1 40 %

Create an HTML form that requires the following information:

1. surname;
2. email;
3. phone number;
4. the size of any deposit;
5. the length of time that the user wishes to take out the loan for: allowed values are 10, 20 and 30 years.
6. annual salary in pounds;
7. how much the user wishes to borrow;
8. knowledge of if the user has an account with the building society (as this may allow a preferential deal).

Basic client side data validation is performed on all of the above data, and errors are reported to the client in an simple manner. Details of tests are recorded in test-harness.pdf.

0.1.1.1 ☞:

We assume that the user will be truthful about their salary and if they have an account with the building society, the software is just a guide to give an indication of how much they can borrow. A real world example is available her <https://www.hsbc.co.uk/1/2/mortgages/how-much-can-i-borrow>
There is no need for any server-side communication to confirm the client has a bank account, a sufficient deposit *etc.*, we are just creating a helpful tool.

	interest rate	10 years	20 years	30 years	maximum loan value	deposit	salary	account required?	amount repayable	monthly repayment
Option A	$x+0.8$	✓	✓	✗	$4s$	$d \geq 0$	s	n		
Option B	$x+0.7$	✓	✓	✗	$4.1s$	$d \geq 0$	s	y		
Option C	$x+0.6$	✗	✓	✗	$5(s+d)$	$d \geq 10000$	s	n		
Option D	$x+0.4$	✓	✗	✓	$6(s+d)$	$d \geq 20000$	s	n		
Option E	$x+0.2$	✓	✓	✗	$7(s+d)$	$d \geq 40000$	s	y		

Table 1: For brevity salary= s , deposit= d , the interest rate $x = 3\%$.

0.1.2 40-49 %

If and only if the data is correct then the possible mortgage offers are displayed. these should be displayed at the bottom of the form. So that the user can (for example) modify their deposit value and see the effect on the mortgage offers instantly. At this level the software need only state the allowed options.

0.1.2.1 data example

For clarity, consider a user who provides the following data:

1. 20 year mortgage;
2. no deposit;
3. salary of £40000;
4. holds an account with the building society;
5. loan amount £100000.

The two options (given he has no deposit and $4s > 100000$) are A and B and the application should provide him with the important information in an uncluttered manner, see fig. 1. Regarding the interest model we use a simple model given by

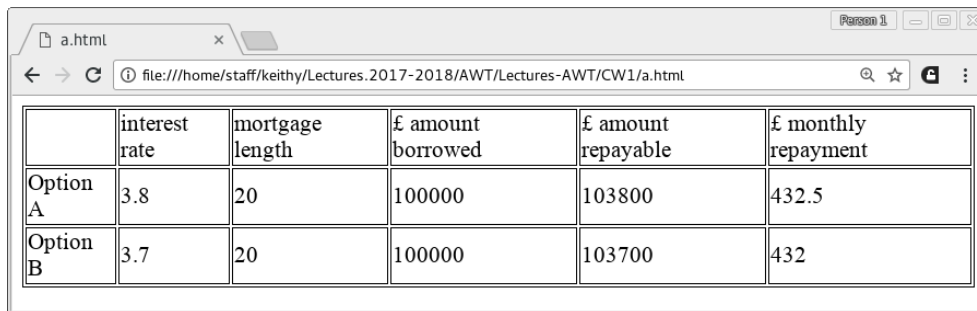
$$\text{amount repayable} = \text{amount payable} \times (1+x) \quad (0.1.1)$$

where x is the the interest rate. For example, borrowing 100000 at 3.8 % costs

$$103800 = 100000 \times 1.038. \quad (0.1.2)$$

0.1.3 50-59 %

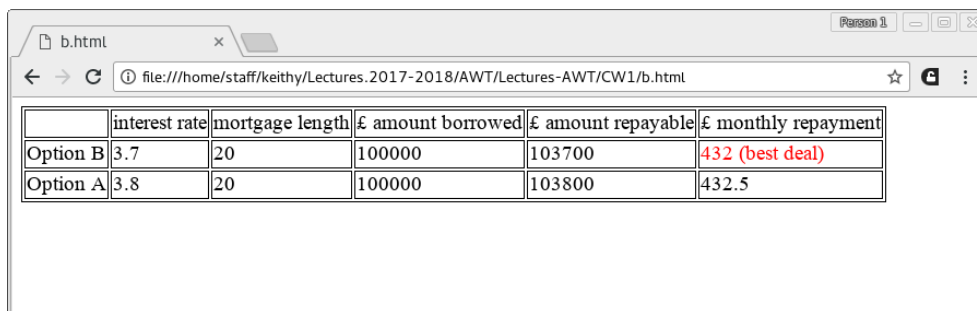
1. At this level Option A and Option B needs to be implemented correctly. Tests should be performed and documented in test-harness.pdf



The screenshot shows a web browser window with a single tab titled 'a.html'. The address bar displays the file path: file:///home/staff/keithy/Lectures.2017-2018/AWT/Lectures-AWT/CW1/a.html. Below the browser interface is a table with 6 columns: an empty header cell, 'interest rate', 'mortgage length', '£ amount borrowed', '£ amount repayable', and '£ monthly repayment'. There are two data rows: 'Option A' with values 3.8, 20, 100000, 103800, and 432.5; and 'Option B' with values 3.7, 20, 100000, 103700, and 432.

	interest rate	mortgage length	£ amount borrowed	£ amount repayable	£ monthly repayment
Option A	3.8	20	100000	103800	432.5
Option B	3.7	20	100000	103700	432

Figure 1: Information displayed to the user when the data in section 0.1.2.1 is used. Only the options that are allowed are shown, as the user has no deposit only Option A and Option B are available.



The screenshot shows a web browser window with a single tab titled 'b.html'. The address bar displays the file path: file:///home/staff/keithy/Lectures.2017-2018/AWT/Lectures-AWT/CW1/b.html. Below the browser interface is a table with 6 columns: an empty header cell, 'interest rate', 'mortgage length', '£ amount borrowed', '£ amount repayable', and '£ monthly repayment'. There are two data rows: 'Option B' with values 3.7, 20, 100000, 103700, and 432 (best deal); and 'Option A' with values 3.8, 20, 100000, 103800, and 432.5. The value '432 (best deal)' is highlighted in red.

	interest rate	mortgage length	£ amount borrowed	£ amount repayable	£ monthly repayment
Option B	3.7	20	100000	103700	432 (best deal)
Option A	3.8	20	100000	103800	432.5

Figure 2: The options are ordered so that the ‘best deal’ (we will define the ‘best deal’ to be the option with the smallest repayment per month) is show first in the table and emphasised in some manner.

2. The allowed options should be displayed in an intuitive HTML table style, for example with the data in section 0.1.2.1 then the user should see something similar to fig. 1.

0.1.4 60-69 %

1. All options A, B , C , D and E need to be implemented correctly. Tests should be performed and documented in test-harness.pdf.
2. The form is dynamic in the following manner: if the software detects an error in a field then it will not allow data entry into the form elements after it until the issue in the field is addressed.
3. If the data is correct then the allowed options are displayed and ordered in such a way that the option with the smallest monthly repayment is displayed first in the table and is emphasised in some manner, see fig. 2

0.1.5 70-79 % - introducing a server

1. Currently the data in table 1 is based upon the value x , which is fixed at 3%. In reality mortgage interest rates follow the dynamic variable which is the bank of England's base rate. Modify your code so that the variable x is assigned dynamically; that is the application needs to find the information on the web retrieve it and assign the value to x . Note this is the first point at which the application becomes a client-server application.
2. A description of the code that resolves 1 is in test-harness.pdf

0.1.6 80+ %

Reflect upon what the software currently does:

1. In test-harness.pdf state, with reasons, additional features that you think would aid the user of the software.
2. Implement the features that you discussed in point 1 above.

0.2 Submittal

To be submitted onto moodle by 01 Dec 17 a zip file that contains the application and the pdf harness. Regarding naming conventions, if i did this assignment I would submit Yates-K-01900300.tar.gz, and there would be a pdf test-harness-Yates-K-01900300.pdf in it. The contents of the pdf should be split into sections corresponding to each degree classification. In each degree classification section provide evidence via tests, tables and figures to indicate that your software functions at the level stated. The report should devote two to three pages to each degree classification (if that classification level was attempted).

0.3 Integrity of Submission

In the lab sessions following the submission deadline I will download your assignment and run it. You will demonstrate the work on a one-to-one basis with me and answer some simple questions on it. This ensures three things:

1. You get to explain what your software does and does not do.
2. I am sure the work is yours, you are referred to MMU regulations regarding plagiarism
3. I will be able to give you feedback on your work and give some indication of its mark (I will look at all the submissions again outside the lab session environment).