**Heartbeats**

In this activity, you’ll make a program that processes real ECG (electrocardiogram) data from a medical database. Your program will go over the data and detect heartbeats.

Task .

**Step 1**

**Open** this [program](http://ncce.io/py-ecg-1) (ncce.io/py-ecg-1) in your development environment.

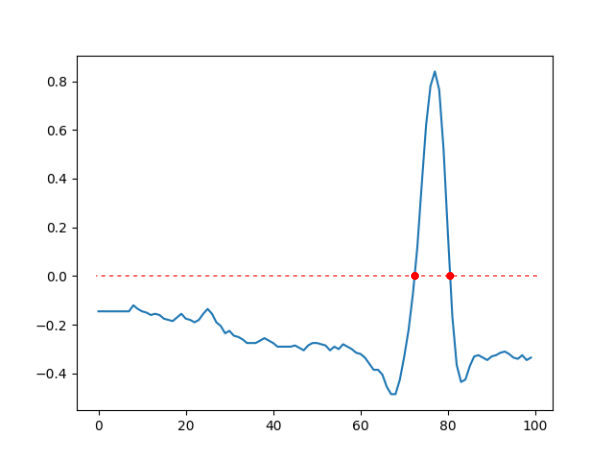
|  |  |
| --- | --- |
| 1  2  3 | from ncce.mitdb\_data import load, plot  heartbeat\_data = load(**100**)  plot(heartbeat\_data, 'heartbeats.png') |

Line 1 imports the load and plot functions from the mitdb\_data module. This is **not a standard Python component**. It has been created specifically to allow you to perform these tasks.

The first 100 values will be loaded from the dataset (this is the highlighted 100 in line 2).

**Step 2**

**Run** the program. It will create a plot of the loaded values in heartbeats.png.

****

As you can see, the values are numerical and can range from -1.0 to 1.0. What you see in this plot of the first 100 values is a single **heartbeat**: values steadily rising over zero, reaching a peak, and then smoothly dropping below zero again.

**Step 3**

One way to detect a heartbeat is to look for **zero crossings** in the data (marked with red dots on the image above). A zero crossing is a point where values change from positive to negative (or vice versa).

**Add** the following **incomplete** code to your program, which will iterate over every value in the data, making sure (in the last line) that the previous value is also available.

|  |  |  |  |
| --- | --- | --- | --- |
| +  +  +  +  +  + | previous = -1.0  for value in heartbeat\_data:   |  |  | | --- | --- | |  | Check for crossing . |   previous = value |

**Step 4**

**Complete** the missing instructions in your program (the labeled box), so that your program prints the message "heartbeat detected" every time it runs across a value that is positive and its previous value is negative.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. This is the output your program should produce for the first 100 values from the data set. | |
| The program displays the message once, i.e. there is one heartbeat in the data loaded. | heartbeat detected |

**Step 5**

**Modify** line 2 in your existing program, so that 1000 data values are loaded from the dataset, instead of 100.

|  |  |
| --- | --- |
| 2❇ | from ncce.mitdb\_data import load, plot  heartbeat\_data = load(**1000**)  plot(heartbeat\_data, 'heartbeats.png') |

**Step 6**

**Run** your program. How many heartbeats is it detecting?

|  |
| --- |
|  |

If you want to know if the number of heartbeats detected is correct, check the updated plot of the loaded values in plot.png. You can simply see how many heartbeats are contained in the data.

Explorer task .

**Step 1**

**Modify** the existing program so that it **counts** the number of detected heartbeats, instead of displaying a message whenever a heartbeat is detected.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. This is the output your program should produce for the first 1000 values from the data set. | |
| The program displays the number of heartbeats detected. | 4 heartbeats detected |

**Step 2**

The data contains **360 values per second**. **Modify** line 2 so that 15-minutes worth of values are loaded from the dataset.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. This is the output your program should produce for 15-minutes worth of data. | |
| The program displays the number of heartbeats detected in 15 minutes. | 1116 heartbeats detected |

**Step 3**

**Extend** the existing program, so that it calculates and prints the **average number of heartbeats per minute** for these 15 minutes.

|  |  |
| --- | --- |
| **Example** |  |
| Note: Use this example to check your program. This is the output your program should produce for 15-minutes worth of data. | |
| The program displays the average number of heartbeats per minute. | Average heart rate is 74.4 |

This resource is licensed under the Open Government Licence, version 3. For more information on this licence, see [ncce.io/ogl](http://ncce.io/ogl).