Task-by-Task Guide

If you would like a little more support while completing this project, explore this step-by-step resource to get additional hints and resources to help you along each task.

Task 1 – Import required libraries

For this task you will gather the required libraries to include in your application to write the code.

Here, we are importing the following required libraries: csv for writing data to a CSV file, datetime for getting the current date, requests for sending HTTP requests to the website, BeautifulSoup for parsing the HTML source code of the webpage, and time for introducing a delay in our program.

This is usually done at the top of the file, for example:
from datetime import datetime

Resources:

importing modules from a package -- Python.org

Task 2 – Generating a URL with a function.

Now that you have the required imports, you need to define a function that takes in two parameters: position and location. It is useful to define a function to allow for the generation of required data, in this case to form a URL for the web scrape request. If the parameters change, you can change the one function rather than changing the code in multiple places.

The parameters in this function are needed to generate the URL of the webpage we want to scrape. Using a function allows you to use a template URL and replace the placeholders with the actual parameter values of position and location. For example, the URL may also include some additional parameters such as locT=C and locId=1139970 that specify the location of the job posting. You can customize these parameters based on your needs.

Resources:

Python Functions -- W3Schools
Decipher the URLs -- RealPython

Task 3 – Extract the Job Data from a single job posting card.

The next step is to define a function that will take a single job posting record as input and extract the relevant data from it. The job posting is a Beautiful soup object. This function will be called from within the main() function, which you will define in the next step.

To do this, we'll use the BeautifulSoup object to parse the HTML of the individual job posting and extract the desired data using a series of try/except blocks to protect the program and the provide the data with known values in case some data is missing from the posting. For example:

```
try:
    job_title = atags[0].text.strip()
except IndexError:
    job_title = ""
```

It is important to show that you use functions to break problems down into constituent parts to make your program more efficient and maintainable.

Resources:

Beautiful Soup -- GeeksForGeeks

Task 4 – Define the main function.

Every program needs a starting point, and a Python application is no different. Define the main function that takes two parameters: job position and location. This function performs the following steps:

1. Set the headers for the HTTP request. A website may block requests from bots, so it's a good idea to set a user agent string.
2. Construct the URL for the job search based on the job position and location using the function you created earlier.
3. Send an HTTP request to the URL and retrieve the HTML code of the search results page.
4. Parse the HTML code using BeautifulSoup and select the HTML elements that contain the job postings (hint: use the Beautiful Soup's findall method).
5. For each posting, extract the job posting information using the helper function from task 3 and store it in a list.
6. Write the job posting information to a CSV file.
7. Print a success message.

Run the program by calling the main function with the required parameters. For example in Jupyter Notebook: main('developer', 'texas').

Check the CSV file to verify that the job posting information has been extracted correctly.

Resources:

Calling Functions -- W3Schools

Beautiful Soup Findall -- Geeks-for-geeks

Task 5 – Describe Conclusions

Write a conclusion about your process and any key findings.

This is your opportunity to impress your prospective employer with your critical thinking and problem-solving skills. You may want to discuss the process you followed and share your struggles and how you overcame them. What do you think sets your portfolio project apart from those of other candidates?

You may even want to offer ideas for improving the design for future business endeavors.

At this point, you can prepare the project artifacts for uploading into your portfolio. You should include:

- An image file of your CSV file with all the fields and data displayed.
- Excerpts from your code explaining the purpose of each function.
- Any improvements/changes you would make to the application.