Congratulations! You passed!

Grade received 100%

Latest Submission Grade 100%

To pass 80% or higher

Go to next item

1. When it is a good opportunity to write a single file Python script? **1 / 1 point**

● When automating a task that doesn't need many inputs and it can work without dependencies

○ When automating a task that needs to be shared and distributed to other systems

○ When automating a task that can use the Click framework for parsing options

✓ **Correct**
Correct. A single file script that doesn't need dependencies is a quick way to automate.

2. What is a module from the Python standard library you can use to look at arguments passed into a script? **1 / 1 point**

● `sys.argv`

○ `os.argv`

○ `os.args`

✓ **Correct**
Correct!

3. When is it a good time to use a command-line tool framework like ArgParse? **1 / 1 point**

● When wanting to handle different options, values, and flags

○ When wanting to distribute the script to other systems and users

○ When packaging a script for publishing

✓ **Correct**
Correct. When multiple options, values, and flags need to be handled, it is easier to use a framework rather than parsing them yourself.

4. What is a good way to use external dependencies in a command-line tool? **1 / 1 point**

○ By creating a Python package and including the dependencies in the same package

● Declaring them in a `setup.py` file that can optionally read them from `requirements.txt`

○ By declaring them in a `README.md` file as well as the help menu of a tool, so that users can install the dependencies needed.

✓ **Correct**
Correct! Using packaging tools like `setup.py` will allow you to declare dependencies so that they are automatically installed in user's system.

5. When packaging a Python command-line tool, does the code need to be in a directory? Or can it be a single file? **1 / 1 point**

● It can be both. There is no hard requirement when packaging a tool.

○ The code for a command-line tool can be in a single file. A directory is only useful for complex services.

○ A command-line tool must always go in a directory so that other supporting files can be added.

✓ **Correct**
Correct!

6. What are some of the key differences between the Click framework and the Argparse framework for building command-line tools?

1 / 1 point

● The Click Python framework is designed to be more user-friendly than the argparse framework. It also offers more features, and it is easier to extend.

○ The argparse framework is designed to be more user-friendly than the Click framework. It also offers more features, and it is easier to extend.

○ The Argparse framework uses decorators primarily for declaring and adding options where Click uses objects and classes.

✓ **Correct**
Correct!

7. What are some benefits of using GitHub Actions?

1 / 1 point

○
1. Ability to deploy easily
2. Easily create and share internal workflows
3. Integrate with other services
4. Access to a community of users

○
1. Ability to replace tools like Puppet, Chef, or Ansible
2. Elasticity: ability to scale up or down as needed depending on demand
3. You can use GitHub Actions for a variety of purposes like storage, computing, or even networking
4. Access to a community of partners

●
1. Ability to automate tasks
2. Easily create and share workflows
3. Integrate with other services
4. Access to a community of users

✓ **Correct**
Correct!

8. What are two benefits of using the Flask framework? **1 / 1 point**

○
1. It includes a powerful database library (Object Relational Mapper) to interact with the database easily.
2. It is well supported by the community as one of the most popular frameworks for Python

●
1. It is an easy framework to use for building HTTP APIs
2. It is well supported by the community as one of the most popular frameworks for Python

○
1. It is an easy framework to use for building HTTP APIs
2. It comes with an optional admin interface for interacting with the database directly

✓ **Correct**
Correct!

9. What is one good reason to use the FastAPI framework? **1 / 1 point**

● It uses type hints that allow to detect common errors in HTTP APIs when handling requests

○ It has one of the fastest request-handling processes which increases performance when deployed

○ It makes it very easy to handle multi-threaded requests for increased performance

✓ **Correct**
Correct!

10. What are common usages of **`POST`** **`GET`** and **`PUT`** HTTP requests? **1 / 1 point**

○
1. **`POST`** is for updating an existing resource
2. **`GET`** is for read-only requests, usually to update data
3. **`PUT`** is to download and existing resource

●

1. **`POST`** is for creating a resource
2. **`GET`** is for read-only requests, usually to retrieve data
3. **`PUT`** is to update an existing resource

○

1. **`POST`** is for updating an existing resource
2. **`GET`** is for read-only requests, usually to read data
3. **`PUT`** is to create a new resource

✓ **Correct**

Correct!