Directory Operations Solutions

directory_entry

- Briefly describe std::filesystem's directory_entry
  - directory_entry is a class that represents an entry in a directory
  - It has a member which is the entry's path
  - It can be converted to path
  - All the operations which can be performed on a path object can be performed on a directory_entry object

directory_iterator

- Briefly describe std::filesystem's directory_iterator
  - directory_iterator is a class that can be used to iterate over all the entries in a directory
- What happens if the directory contains a subdirectory?
  - The subdirectory will have an entry in the directory
  - The entry for the subdirectory will be included in the iteration
  - However, any entries in the subdirectory will not be included
- Write a program that uses directory_iterator to display the names, properties and permissions of all the files in a directory

Creating a Directory

- Write down a statement that creates a new directory using std::filesystem
  - create_directory("temp");
- What happens if the directory already exists?
  - Nothing. The operation silently fails
- What are the default permissions on the new directory?
  - All users can perform all operations
- How can these permissions be changed?
  - If we have an existing directory with the required permissions, we can pass this as a second argument
  - The new directory will have the same permissions as the one in the argument

Creating a Directory

- Write a program that
  - Creates a directory and some files in it
  - Iterates over the entries in the directory
  - Displays the properties and permissions of all the entries

Creating Multiple Directories

- Write down a statement that uses std::filesystem to create a path that contains a sequence of new directories
  - create_directories("temp/temp1/temp2/temp3");
- What happens if some of the "new" directories already exist?
  - Nothing. Any existing directories are ignored

remove_all()

- Briefly describe the remove_all function in std::filename
  - remove_all() takes a path as argument
  - It will delete a directory and all the entries in it
  - This includes subdirectories and all their entries

recursive_directory_iterator

- Briefly describe std::filesystem's recursive_directory_iterator
  - recursive_directory_iterator is similar to directory_iterator
  - However, if it encounters a subdirectory, it will iterate over the subdirectory's entries before continuing
- Write a program that
  - Creates a directory and some files in it
  - Creates a sequence of new directories in the new directory
  - Recursively iterates over the entries in the directory
  - Displays the properties and permissions of all the entries

current_path()

- Briefly describe std::filesystem's current_path
  - When called with no arguments, current_path() gets the program's current working directory
  - When called with a path argument, it will set the current working directory to that path
- Write a program which
  - Creates a directory and some files in it
  - Sets the new directory to be the working directory
  - Displays the names of all the files in this directory
  - Changes the working directory back to its original value

Disk Space

- Briefly describe how to use std::filesystem to find the amount of storage on a disk
  - std::filesystem's space() returns a space_info struct
  - space_info has three fields
  - capacity is the total amount of storage on the disk
  - free is the amount of unused storage on the disk
  - available is the amount of storage available to non-privileged users
- Write a simple program which displays the amount of storage on your computer's disk in gigabytes