Assignment based on shell programming

LIST OF PROBLEMS
---

1. List the subdirectories/files of given directory(or directories). The user can specify whether dir only or file only listing is desired as an option. In either case the number of files/dirs is also to be displayed.

2. List the files of specified directory (or directories). The user can specify as an option what type of files he/she wants, such as executable files/read-only files/setuid files, etc. The number of such files found are also to be displayed.

3. List all the binary files (executable) in a specified directory (or directories). In this problem you have to find all the files (executable) that exist in the file system tree below the directory specified (examine the subdirectories also). These files should be displayed sorted on size (decreasing order) and the number of such files should also be given at the end.

4. Given two integers indicating lower and upper range value, generate all even integers in this range. Through an option, a user may specify how many numbers be displayed in a single line (1 to 5).

5. List all files that have size larger than a specified size. The output should be sorted on size, in decreasing order. A directory (or directories) would be specified as input and all files in this directory (including its subdirectories may be) should be examined for finding the desired information.

6. Write a program to support delete and undelete facility on linux.

7. Write a program that makes multiple copies of a specified file. The user may specify the names of all the copies in a suitable manner. The idea is to have a more general version of cp.

8. Choose two algorithms for solving the same problem, for example, one may choose heapsort and hashing for solving a search problem where one the number of elements are unordered and large, say 100000 elements.

   - Compile and produce two executables for the two algorithms after ensuring that the implementations are correct.
   - Write a program that can generate input files when the relevant parameters are supplied as input. Use random generators in your
   - Write a program in C/C++ that cn reads the timing stats given in a file to determine which algorithm is more efficient in practice.
   - Write a shell program that glues all the executables produced above to produce results in favour of the better algorithm.