Read Me Assignment 1 Finds, Regex Engine, Printf

WildCard Feature Documentation:

The wildcard functionality in the "finds" program supports all specified wildcards (., *, ?) along with parentheses for grouping. This feature allows wildcards inside parentheses as well. To test this functionality, create a file containing various strings, then use the following command format:

```
./finds -p <pathname> [-f c|h|S] [-l] -s <search_pattern>
```

Ensure that the search string is enclosed in quotes to prevent the terminal from interpreting wildcards as commands. For example, to search within a specific directory, you might use:

```
./finds /home/Desktop/Test -f c -s "AB(CD)*EFF"
```

Utilize the `make` command to compile all files listed in the Makefile and `make clean` to remove compiled files before running any program.

Edge Cases/Limitations:

- Passing an empty string "" will always result in a match, as an empty string is considered to match any content.
- Wildcards within parentheses are fully supported except for certain scenarios involving the `*` wildcard within parentheses, which may occasionally lead to segmentation faults. For example, the search pattern e(a*)*d might not work as expected.
- Mixing and matching wildcards is not allowed, adhering to the assignment requirements.

Additional Notes:

Invalid character inputs will trigger an error message in the terminal, specifying the character and providing a relevant message. While the wildcard functionality has been extensively tested and works for all specified wildcards and parentheses, if you encounter unexpected errors during testing, consider bypassing the `isValidRegex` function in `searchFile.c`. This function may inadvertently return an error for valid wildcard patterns. The implementation successfully handles symlinks, printing both the symlink name and the target file name, which results in duplicate outputs—one for the symlink and another for direct file access. When creating symLinks to test functionality please use absolute paths. **The bonus for wildcard functionality has been completed.**

Printf Feature Documentation:

A custom printf function, compatible with the format specifiers %s, %c, %d, %u, and %x, is included in `myPrint.c`. This custom function does not utilize `va_list`, `va_start`, `va_arg`, or `va_end` for handling variable arguments, as per the assignment constraints. Instead, it employs an alternative method for variable argument management. To test this function, run it in a C compiler with the provided test cases. For convenience, you can use the following online compiler:
[https://www.programiz.com/c-programming/online-compiler/]. Copy and paste the code from `myPrint.c` into the compiler and run the included test cases. To test for other cases just try different cases inside the main! **The bonus for the `printf` functionality has not been completed.**