Dokumentacja projektu "myBash"

Opis

Projekt "myBash" jest prostym interpreterem poleceń w stylu powłoki Bash, napisanym w języku C. Interpreter obsługuje podstawowe polecenia oraz potoki (pipes), umożliwiające przekierowywanie strumienia danych z jednego procesu do drugiego.

Sposób uruchamiania

Aby uruchomić interpreter "myBash", należy skompilować plik źródłowy za pomocą kompilatora języka C, na przykład:

```
gcc myBash.c -o myBash
```

Następnie można uruchomić interpreter bez argumentów, co spowoduje uruchomienie go w trybie interaktywnym, lub przekazać skrypt jako argument na przykład:

```
./myBash commands.txt
```

Sposób testowania

Interpreter "myBash" można testować na różne sposoby:

1. **Testy jednostkowe**: Można napisać testy jednostkowe, które sprawdzają zachowanie poszczególnych funkcji w izolacji.
2. **Testy integracyjne**: Można stworzyć skrypty testowe, które wykorzystują interpreter "myBash" do wykonania różnych złożonych poleceń i sprawdzenia poprawności ich wykonania oraz przekierowania strumieni danych.
3. **Ręczne testowanie**: Można uruchomić interpreter interaktywnie lub z plikiem poleceń i przetestować jego zachowanie w różnych sytuacjach.

Zaimplementowane funkcjonalności

1. **Wykonanie poleceń**: Interpreter "myBash" umożliwia wykonanie podstawowych poleceń systemowych, takich jak ls, pwd, cat, itp.
2. **Obsługa potoków (pipes)**: Możliwe jest tworzenie potoków poleceń, które przekierowują strumienie danych z jednego procesu do drugiego.
3. **Obsługa przekierowań do plików**: Interpreter umożliwia przekierowywanie wyniku działania poleceń do plików przy użyciu operatora >>.

4. **Obsługa komend w tle**: Interpreter umożliwia uruchamianie poleceń w tle, dodając znak & na końcu polecenia. W takim przypadku interpreter nie będzie czekał na zakończenie wykonania komendy, tylko od razu przejdzie do kolejnej.
5. **Historia poleceń**: Każde wykonane polecenie jest zapisywane w pliku .myBash_history.txt i może być wyświetlone sygnałem SIGQUIT.
6. **Obsługa sygnałów**: Interpreter reaguje na sygnał SIGQUIT, wyświetlając historię wykonanych poleceń.
7. **Obsługa skryptów**: Interpreter umożliwia uruchamianie skryptów, co pozwala na wykonywanie serii poleceń zapisanych w pliku tekstowym. Skrypty mogą zawierać specjalne polecenia oraz komentarze, a ich wykonanie kończy się po wykonaniu wszystkich poleceń zawartych w skrypcie lub po otrzymaniu znaku końca pliku.

Przykładowa funkcja

*void history(char *command)*

Funkcja ta zapisuje wykonane polecenie do pliku historii .myBash_history.txt. Jeśli plik nie istnieje, zostanie utworzony. Jeśli liczba poleceń w historii przekroczy 20, starsze wpisy zostaną usunięte, aby zachować ograniczoną historię.

Sposób użycia:

```
char command[100] = "ls -l";
history(command);
```

Ta funkcja zapisze polecenie "ls -l" do pliku historii, przechowując go w .myBash_history.txt.

Skrypt testujący skrypt.myBash

W załączonym archiwum znajduje się przykładowy skrypt testujący dla interpretera "myBash", nazwany skrypt.myBash. Skrypt ten wykonuje różne polecenia, sprawdzając zachowanie interpretera w różnych sytuacjach.

Ten skrypt wykonuje następujące czynności:

1. Wykonuje listowanie plików w bieżącym katalogu, przekierowuje wynik do polecenia grep, a następnie przekierowuje wynik do pliku asd.
2. Wykonuje ponowne listowanie plików w bieżącym katalogu i przekierowuje wynik do pliku asd2.
3. Czeka 7 sekund.
4. Usuwa pliki asd i asd2.
5. Uruchamia polecenie sleep który powininen wstrzymać wykonywanie skryptu ale jest uruchomiony w tle, więc od razu przechodzi do punktu 6.

6. Tworzy plik nowy.txt.
7. Czeka 2 sekundy.
8. Wysyła sygnał SIGQUIT do procesu myBash, aby wyświetlić historię wykonanych poleceń.
9. Czeka 2 sekundy.