Zarządzanie pamięcią, biblioteki, pomiar czasu

Zadanie 1. Alokacja tablicy ze wskaźnikami na bloki pamięci zawierające znaki (25%)

Zaprojektuj i przygotuj zestaw funkcji (bibliotekę) do zarządzania tablicą bloków, w których to blokach pamięci zapisywane są rezultaty operacji zliczania lini, słów i znaków (poleceniem *wc*) w plikach przekazywanych jako odpowiedni parametr.

Biblioteka powinna umożliwiać:

- utworzenie tablicy wskaźników w której będą przechowywane wskaźniki na bloki pamięci zawierające wyniki
- przeprowadzenie zliczenia lini, słów i znaków dla zadanych plików i zapisanie wyniku zliczania w pliku tymczasowym
- zarezerwowanie bloku pamięci o rozmiarze odpowiadającym rozmiarowi pliku tymczasowego i zapisanie w tej pamięci jego zawartości, ustawienie w tablicy wskaźników wskazania na ten blok, funkcja powinna zwrócić indeks stworzonego bloku w tablicy,
- usunięcie z pamięci bloku o zadanym indeksie

Tablice / bloki powinny być alokowane przy pomocy funkcji `calloc()` (alokacja dynamiczna).

Przygotuj plik *Makefile*, zawierający polecenia kompilujące pliki źródłowe biblioteki oraz tworzące biblioteki w dwóch wersjach: statyczną i współdzieloną.

Zadanie 2. Program korzystający z biblioteki (25%)

Napisz program testujący działanie funkcji z biblioteki z zadania 1.

Jako argumenty przekaż liczbę elementów tablicy głównej (liczbę par plików) oraz listę zadań do wykonania. Zadania mogą stanowić zadania zliczenia dla wszystkich plików lub zadania usunięcia bloku o podanym indeksie.

Operacje mogą być specyfikowane w linii poleceń na przykład jak poniżej:

- `create_table rozmiar` — stworzenie tablicy o rozmiarze "rozmiar"
- `wc_files file1.txt file2.txt …` — zliczenie dla plików
- `remove_block index` — usuń z tablicy bloków o indeksie *index*

Program powinien stworzyć tablice bloków o zadanej liczbie elementów

W programie zmierz, wypisz na konsolę i zapisz do pliku z raportem czasy realizacji podstawowych operacji:

- Przeprowadzenie zliczeń plików — różna wielkość plików (małe, średnie, duże) oraz różna ilość plików na raz (1 - 10)
- Zapisanie, w pamięci, bloków o różnych rozmiarach (odpowiadających rozmiarom różnych przeprowadzonych zliczeń)
- Usunięcie zaalokowanych bloków o różnych rozmiarach (odpowiadających rozmiarom różnych przeprowadzonych zliczeń)
- Na przemian kilkakrotne dodanie i usunięcie zadanej liczby bloków

Mierząc czasy poszczególnych operacji, zapisz trzy wartości: czas rzeczywisty, czas użytkownika i czas systemowy. Rezultaty umieść pliku *raport2.txt* i dołącz do archiwum zadania.

Zadanie 3. Testy i pomiary (50%)

a. (25%) Przygotuj plik *Makefile*, zawierający polecenie uruchamiania testów oraz polecenia kompilacji programu z zad 2 na trzy sposoby:
    - Z wykorzystaniem bibliotek statycznych,
    - Z wykorzystaniem bibliotek dzielonych (dynamiczne, ładowane przy uruchomieniu programu),
    - Z wykorzystaniem bibliotek ładowanych dynamicznie (dynamiczne, ładowane przez program).

b. Wyniki pomiarów zbierz w pliku results3a.txt. Otrzymane wyniki krótko skomentuj.

c. (25%) Rozszerz plik *Makefile* z punktu 3a) dodając możliwość skompilowania programu na trzech różnych poziomach optymalizacji — `-O0`…`-Os`. Przeprowadź ponownie pomiary, kompilując i uruchamiając program dla różnych poziomów optymalizacji.
   Wyniki pomiarów dodaj do pliku results3b.txt. Otrzymane wyniki krótko skomentuj.

Wygenerowane pliki z raportami załącz jako element rozwiązania.

Uwaga: Do odczytania pliku można użyć funkcji `read()` (`man read`), do wywołania zewnętrznego polecenia Unixa można użyć funkcji `system()` (`man system`).