

**Lab 11. Struktury danych cd.: porządkowanie struktur. Obsługa plików binarnych.**

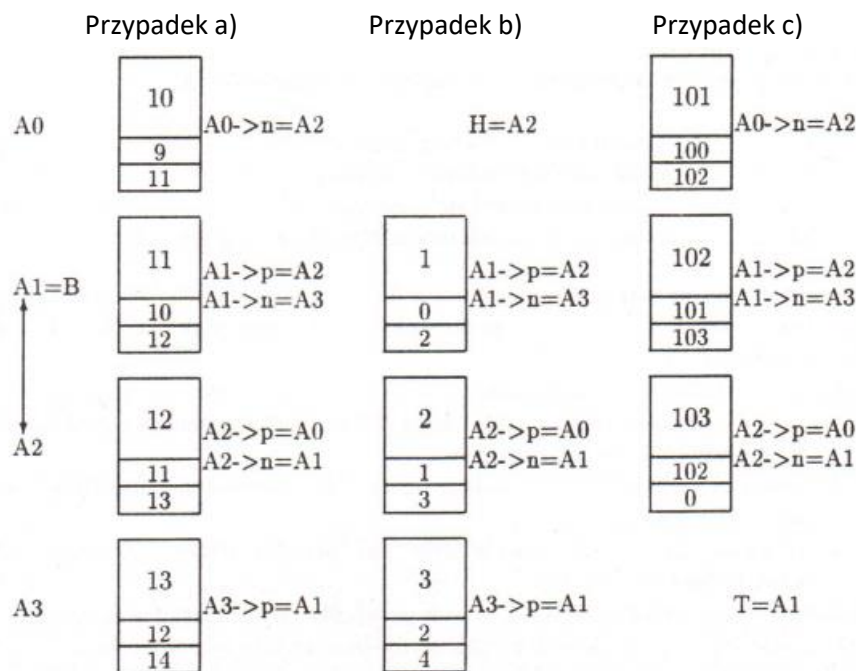
- W pliku tekstowym zapisane są następujące dane o wielu studentach:  
imię, nazwisko, rok urodzenia, adres zamieszkania, kwota stypendium.  
Bazując na strukturach zaimplementuj listę dwukierunkową (program z lab. 10). Zbuduj funkcję, która uporządkuje ciąg struktur malejąco wg kwoty stypendium. Wykorzystaj algorytm sortowania bąbelkowego.

Przyjmij następujące oznaczenia:

$A0 = B \rightarrow p$ ;    wskaźnik do struktury poprzedniej  
 $A1 = B$ ;        wskaźnik do struktury bieżącej  
 $A2 = B \rightarrow n$ ;    wskaźnik do struktury następnej  
 $A3 = A2 \rightarrow n$ ;    wskaźnik do struktury następnej względem A2

Rozpatrz trzy przypadki:

- Żadna ze zmiennych A0, A1, A2, A3 nie jest wskaźnikiem ani do struktury pierwszej, ani do ostatniej,
- Zmienna A1=B jest wskaźnikiem do pierwszej struktury, wtedy  $A0 = (\text{STUDENT} *) 0$ ,
- Zmienna A1=B jest wskaźnikiem do przedostatniej struktury, wtedy  $A3 = (\text{STUDENT} *) 0$ .



- Uporządkuj listę struktur alfabetycznie wg nazwisk.
- Wstaw strukturę do uporządkowanej listy struktur, tak by porządek został zachowany.
- Zmodyfikuj funkcję porządkującą struktury tak, by kryterium porządkowania mogło być dowolne.

2. Oblicz iloczyn skalarny dwóch wektorów  $dot = x^T y = \sum x_i y_i$  ( $i \in [0, n-1]$ ). Wektory przedstawić w postaci tablicy struktur danych `MY_VECTOR *tab`; która zawiera 2 elementy: `tab[0]` – wektor `x`, `tab[1]` – wektor `y`. Użyć dynamiczne alokowanie pamięci.

```
struct MY_VECTOR
{
    int dim;           //rozmiar wektora
    char str[64];      //nazwa wektora
    double *buff;      //tablica dla przechowywania wektora o rozmiarze dim
};
```

Obsługa wektorów powinna zawierać:

- Inicjowanie wektora.
  - Przygotowanie wektora: wprowadzenie rozmiaru zadania `n`, nazwy wektora `str` oraz alokowania pamięci i wypełnienia tablicy `buff`. Dla wektora `x`:  $x_i = 1/(i+1)$ , dla wektora `y`:  $y_i = i+1$ .
  - Algorytm mnożenia.
  - Sprawdzenie poprawności wyniku: dla podanych `x` i `y` iloczyn skalarny  $dot = n$ .
  - Zwolnienie pamięci dla tablicy `buff`.
3. Do zapisu i odczytu danych binarnych z pliku służą odpowiednio funkcje: `fwrite` oraz `fread`. Zapoznaj się z opisem funkcji z dokumentacji MSDN. Pamiętaj, aby przy otwieraniu pliku wybrać odpowiedni tryb pozwalający na pracę w trybie binarnym.
- <https://cpp0x.pl/dokumentacja/standard-C/fwrite/501>
- <https://cpp0x.pl/dokumentacja/standard-C/fread/456>
4. Utwórz przykładową tablicę typu `int`. Zainicjalizuj ją kilkoma wartościami. Zapisz ją do pliku w trybie binarnym. Wyzeruj zawartość tablicy, a następnie wczytaj do niej dane z pliku. Wyświetl zawartość na ekranie.