Lab 2b (7 pkt) statyczne tablice 1-wymiarowe

Funkcja **main** jest przygotowana w **całości** i nie należy jej modyfikować (oczywiście poza zakomentowaniem/odkomentowaniem testowanych fragmentów kolejnych etapów).

Wszystkie etapy należy wykonać **w podanej kolejności**, w których należy **zaimplementować** wszystkie **niezbędne funkcje**.

Przykładowe wyniki działania programu są zamieszczone poniżej treści zadania.

Etap 1 (1,5 pkt)

Należy zaimplementować **2 funkcje**, które **można wywołać** następująco:

```
fill1(tab, n, k);
```

- Funkcja **`fill1`** wypełnia 1-wymiarową tablicę **`tab`** o rozmiarze **`n ( n<=N )`** wartościami **ciągu**, gdzie pierwsze **k** elementów to wartości **1, 2, …, k**, a każdy **kolejny** jest **sumą ostatnich k elementów**. Dla **k=2** otrzymujemy ciąg Fibonacciego.

```
print(tab, n);
```

- Funkcja **`print`** wypisuje na ekran elementy 1-wymiarowej tablicy **`tab`** o rozmiarze **`n ( n<=N )`** w formacie **`[ 1 2 3 5 8 13 ]`**

Etap 2 (1 pkt) *należy wykonać **bezpośrednio** w bieżącej tablicy wartości*

Należy zaimplementować **funkcję**, którą można **wywołać** następująco:

```
fill2(tab, n, a, b);
```

- Funkcja **`fill2`** wypełnia 1-wymiarową tablicę **`tab`** wartościami **n** elementów całkowitych **wylosowanych** z przedziału **`[a,b]=[2,20]`**. Docelowo elementy tablicy powinny być **uporządkowane niemalejąco** (elementy mogą się powtarzać). Do losowania użyj funkcji **rand**.

Etap 3 (1 pkt)

Należy zaimplementować **funkcję**, którą można **wywołać** następująco:

```
k = test(tab, n);
```

- Funkcja **`test`** sprawdza, czy elementy 1-wymiarowej tablicy **`tab`** o rozmiarze **`n ( n<=N )`** tworzą **ciąg Fibonacciego**. **Zwraca** 1, gdy **wykryto ciąg** i wtedy zwraca **iloraz** tego ciągu, **wpp** zwraca 0.

Etap 4 (1 pkt)

Należy zaimplementować **funkcję**, którą można **wywołać** następująco:

```
n3=sum(tab1, n1, tab2, n2, tab3);
```

- Funkcja **`sum`** znajduje **sumę (jak dla zbiorów) elementów** 2 uporządkowanych niemalejąco tablic **`tab1`** oraz **`tab2`** o liczbie elementów odpowiednio **`n1`** oraz **`n2`**. Wynikowa tablica **`tab3`** będąca sumą jest tablicą od razu uporządkowaną wg tej samej relacji (jak tab1, tab2), zwraca **liczbę** elementów **`tab3`** (czyli n1+n2).

Etap 5 (1 pkt) *należy wykonać **bezpośrednio** w bieżącej tablicy wartości*
Należy zaimplementować **funkcję**, którą można **wywołać** następująco:

```
rotation(tab, n, k);
```

- Funkcja **`rotation`** przenosi **`k`** ostatnich elementów tablicy **`tab`** (o rozmiarze **`n`**, **`n<=N`**) w jej początkowy fragment (zachowując pierwotną kolejność elementów w obu częściach

Etap 6 (1,5 pkt) *należy wykonać **bezpośrednio** w bieżącej tablicy wartości*
Należy zaimplementować **funkcję**, którą można **wywołać** następująco:

```
ile_par = pair_single(tab, n);
```

- Funkcja **`pair_single`** porządkuje wartości tablicy tak, by w początkowej części tablicy znalazły się **pary elementów** (w dowolnej kolejności, ale elementy pary obok siebie), a w końcowej elementy pozostałe **bez pary** (w dowolnej kolejności). Funkcja zwraca liczbę par

Przykładowe wyniki działania programu dla srand(5)

```
---------------   ETAP 1 (1,5 pkt)   ---------------

tab1: [  1  2  3  5  8  13  21  34 ]
tab2: [  1  2  3  4  10  19  36  69  134  258 ]

---------------   ETAP 2 (1 pkt)   ---------------
tab3: [  2  5  6  8  17  17  18  19  19  19 ]

---------------   ETAP 3 (1 pkt)   ---------------

 tab1 - ciag Fibonacciego
 tab1 - to NIE jest ciag Fibonacciego

---------------   ETAP 4 (1 pkt)   ---------------

Uporzadkowana suma tab4 dla tablic tab2 i tab3

tab4: [  1  2  2  3  4  5  6  8  10  17  17  18  19  19  19  19  36  69
134  258 ]

---------------   ETAP 5 (1 pkt)   ---------------

Rotacja tab2

tab2: [  1  2  3  4  10  19  36  69  134  258 ]

tab2: [  69  134  258  1  2  3  4  10  19  36 ]

---------------   ETAP 6 (1,5 pkt)   ---------------

Single i pary: tab5

tab5: [  3  2  5  7  3  6  4  7  7  8  2  8  6  6  2  3  2  5  9  1 ]

liczba par: 7
tab5: [  5  5  2  2  7  7  6  6  3  3  8  8  2  2  6  3  4  7  9  1 ]
```

```
Pary i Single: tab6

tab6: [  1  2  3  4  5  1  2  3  4  5  1  2  3  4  5  1  2  3  4  5 ]

liczba par: 10
tab6: [  5  5  2  2  4  4  1  1  3  3  5  5  2  2  4  4  3  3  1  1 ]

Pary i Single: tab7

tab7: [  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19
20 ]

liczba par: 0
tab7: [  1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19
20 ]

KONIEC
Press any key to close this window . . .
```