Laboratorium 2

5. Rozwinięcie dziesiętne

   (a) Dla zadanej liczby wymiernej `p/q`, znajdź jej rozwinięcie dziesiętne (skończone lub okresowe). Zastosuj w rozwiązaniu odpowiednie funkcje.

   Przykład 1
   Dane wejściowe
   `9/20`
   Dane wyjściowe
   `0.45`

   Przykład 2
   Dane wejściowe
   `2/3`
   Dane wyjściowe
   `0.(6)`

   (b) Przedstaw **obiektową** wersję rozwiązania problemu.

6. Liczby doskonałe

   (a) Znajdź `n`-tą w kolejności liczbę doskonałą.
   Zastosuj w rozwiązaniu następujące funkcje:
   - funkcję `czy_podzielnik` badającą, czy zadana liczba `p` jest podzielnikiem `l`,
   - funkcję `czy_doskonala` badającą, czy zadana liczba `l` jest liczbą doskonałą,
   - funkcję `pierwsza_dosk` znajdującą pierwszą liczbę doskonałą,
   - funkcję `wieksza_dosk` znajdującą najmniejszą liczbę doskonałą większą od `d`.

   Przykład

   Dane wejściowe
   `4`

   Dane wyjściowe
   `8128`

   (b) Przedstaw **obiektową** wersję rozwiązania problemu.
   Umieść metody `czy_podzielnik` i `czy_doskonala` w sekcji prywatnej, a metody `pierwsza_dosk` i `wieksza_dosk` w sekcji publicznej klasy.

Zadanie obowiązkowe

7. Sortowanie „300*1000”

   Porównaj eksperymentalnie złożoność trzech algorytmów sortowania: bąbelkowego, stogowego i szybkiego. Użyj/zdefiniuj w tym celu trzy różne funkcje sortowania ciągu liczb całkowitych (`babelkowe`, `stogowe` i `szybkie`), a także uniwersalną funkcję pomiaru (`pomiar`) do realizacji pomiarów dla wskazanego ciągu liczb i wskazanego algorytmu sortowania (parametr przekazany przez wskaźnik funkcji!). Przeprowadź `1000` losowań ciągów `300`-elementowych, o wartościach z przedziału <`0;1000`>. Każdy wylosowany ciąg posortuj trzema algorytmami. Porównaj uśrednione czasy sortowania.
   W komentarzu umieszczonym na końcu programu zapisz wnioski dotyczące zbadanej eksperymentalnie złożoności obliczeniowej.

   Plik wyjściowy powinien przyjąć postać:

   `BA` *t11*
   `ST` *t21*
   `SZ` *t31*

   gdzie *t11* oznacza średni czas sortowania ciągu `300`-elementowego metodą bąbelkową, *t21* – średni czas sortowania stogowego, zaś *t31* – średni czas sortowania szybkiego.

   Nagłówek pomocniczy:

```
int pomiar(float *Dane, int Rozmiar, int Lserii, void(*sort)(float[],int))
```

Zadanie dla chętnych

8. Drzewo genealogiczne

   W pliku wejściowym ‘`drzewo_genealogiczne.txt`’ zadane są relacje macierzyństwa (`m`) i ojcostwa (`o`) pomiędzy członkami pewnej dużej rodziny.
   Relacje zapisano w postaci:

```
(Ewa, m, Maciej)
(Adam, o, Maciej)
```

   Utwórz drzewo genealogiczne (o strukturze niejednorodnej) wiedząc, że nikt z członków rodziny nie ma więcej niż `4` potomków. Znajdź wszystkie prawnuki protoplastów rodu. Zauważ, że w pierwszym kroku trzeba odkryć protoplastów rodu!

   Przykład

   Dane wejściowe

```
(Maciej, o, Jan)
(Maciej, o, Filip)
(Elzbieta, m, Filip)
(Elzbieta, m, Jan)
(Elzbieta, m, Barbara)
(Maciej, o, Barbara)
(Katarzyna, m, Janina)
(Katarzyna, m, Beata)
(Tomasz, o, Janina)
(Tomasz, o, Beata)
(Ewa, m, Maciej)
(Adam, o, Maciej)
(Filip, o, Krzysztof)
(Daria, m, Krzysztof)
```

```
(Filip, o, Antoni)
(Daria, m, Antoni)
(Barbara, m, Waclaw)
(Piotr, o, Waclaw)
(Beata, m, Anna)
(Beata, m Helena)
(Beata, m, Dawid)
(Jerzy, o, Dawid)
(Jerzy, o, Helena)
(Jerzy, o, Anna)
(Waclaw, o, Patrycja)
(Lidia, m, Patrycja)
(Ewa, m, Katarzyna)
(Adam, o, Katarzyna)
```

Dane wyjściowe

```
Krzysztof
Antoni
Waclaw
Anna
Helena
Dawid
```