1. Napisz program obliczający wartość stałej Catalana, zdefiniowanej jako

$$G=\frac{\sum_{n=0}^{\infty}(-1)^n}{(2n+1)^2}=+\frac{1}{1^2}-\frac{1}{3^2}+\frac{1}{5^2}-\frac{1}{7^2}+\frac{1}{9^2}+\ldots$$

Program powinien przyjmować z wiersza poleceń pojedynczą wartość *N* a następnie wypisywać obliczoną wartość stałej Catalana przy sumowaniu dla *n*=0 do *N*−1.

Przykład:

```
catalan 10
0.9169923553509892
```

Sprawdź (w porównaniu np. z Wikipedią — „Catalan's constant"), ile (*N*) należy wziąć wyrazów sumy, aby otrzymać dokładność:

- do 3 cyfr po przecinku
- do 6 cyfr po przecinku
- do 9 cyfr po przecinku

2. Napisz program, który dla każdego parametru podanego w linii poleceń (argc):

- zamieni wszystkie duże litery na małe i odwrotnie
- wypisze parametr na standardowe wyjście

Każdy z parametrów powinien zostać wypisany w osobnej linii. Program może działać tylko na literach alfabetu łacińskiego, nie musi uwzględniać polskich znaków.

Przykład:

```
zamien Ala ma KoTa
aLA
MA
kOtA
```

Wskazówka: zapoznaj się z funkcjami isupper, islower, toupper, tolower (#include <ctype.h>).

3. Napisz program, który ze standardowego wejścia wczytuje trzy liczby *a*, *b* i *c*, a następnie:

- sprawdza czy można zbudować trójkąt o bokach podanej długości
- jeśli tak, oblicza i wypisuje pole jego powierzchni

Przykład:

```
trojkat
3 4 5
TAK
6.0
```

```
trojkat 10 30 50
NIE
```

Wskazówka: zapoznaj się ze wzorem Herona: $P=\sqrt{p(p-a)(p-b)(p-c)}$, gdzie $p=\frac{a+b+c}{2}$.

4. Napisz program, który ze standardowego wyjścia wczytuje współczynniki macierzy 3×3 liczb rzeczywistych, a następnie oblicza jej wyznacznik.

Przykład:

```
macierz
1 2 1
0 1 1
0 0 1
1.0
```

5. Napisz program rozwiązujący równanie $x+e^x=0$ i wypisujący wynik na standardowe wyjście.

Wskazówka: możesz wykorzystać metodę bisekcji:

1. Oznaczmy $f(x)=x+e^x$.
2. Znajdź (np. rysując wykres funkcji) dwie wartości x_L i x_L $(x_L<x_R)$ dla których zachodzi $f(x_L)<0$, $f(x_R)>0$.
3. Dopóki długość przedziału $|x_R-x_L|$ jest odpowiednio duża (np. większa od 10^{-15}), wykonaj
   - oblicz środek przedziału $x=\frac{x_L+x_R}{2}$,
   - jeśli $f(x_L)<0$, zastąp $x_L:=x$, jeśli zaś $f(x_L)>0$, zastąp $x_R:=x$.

Po osiągnięciu sytuacji takiej, że $|x_R-x_L|$ jest bliskie zeru, środek przedziału stanowi rozwiązanie równania.

6. Napisz program pobierający z wiersza poleceń dwa parametry:

- liczbę wierszy W
- liczbę kolumn K

a następnie wypisujący *W* wierszy tekstu, zawierających po *K* kolumn każdy,
gdzie w kolumnie *k* ($1\le k\le K$) i wierszu *w* ($1\le w\le W$) znajduje się losowa litera alfabetu łacińskiego, z tym że:

- litera jest duża jeśli *k* jest nieparzyste i mała jeśli *k* jest parzyste
- litera jest samogłoską jeśli *w* jest nieparzyste i spółgłoską jeśli *w* jest parzyste

Przykład:

```
generuj 3 5
AyUoE
FdJbW
UiOeI
```

Wskazówka: losową liczbę z zakresu od 0 do $N-1$ możesz wygenerować jako `rand() % N`.
Możesz w ten sposób wybierać losowe elementy z predefiniowanej tablicy (gdzie *N* jest rozmiarem tablicy).
Zapoznaj się też z funkcjami tolower, toupper (#include <ctype.h>).

7. Napisz program wykonujący sumowanie ułamków w postaci licznik÷mianownik.

Program powinien wczytywać ze standardowego wejścia linie z ułamkami zapisanymi w postaci licznik/mianownik. Po wczytaniu każdej linii program powinien wypisać sumę wszystkich dotychczas wpisanych liczb, również w postaci ułamka („licznik/mianownik”). Wypisywane ułamki powinny być ułamkami właściwymi, tj. licznik i mianownik powinny być względnie pierwsze.

Program powinien zakończyć działanie gdy standardowe wejście nie będzie zawierać już żadnych ułamków do wczytania (np. użytkownik wciśnie Ctrl+D, czyli znak końca pliku).

Przykład:

```
ulamki
2/6
1/3
1/6
1/2
3/5
11/10
```

Wskazówka: Do skracania ułamka wykorzystaj algorytm Euklidesa. Aby rozpoznać, czy udało się wczytać ułamek, czy też napotkaliśmy koniec pliku, sprawdź rezultat zwracany z funkcji scanf — przy poprawnym wczytaniu zwróci ona liczbę wczytanych zmiennych (np. 1 dla „%d”, 2 dla „%d %d” etc.).

8. Zaimplementuj sortowanie bąbelkowe w postaci analogicznej do deklaracji funkcji qsort z biblioteki standardowej języka C, czyli:

```
void bsort(void* ptr, size_t count, size_t size, int (*comp)(const void*, const void*))
```

gdzie:

- ptr to tablica do posortowania
- count to liczba elementów w tablicy
- size to rozmiar pojedynczego elementu w bajtach
- comp to funkcja do porównywania dowolnych dwóch elementów

Wskazówka: Pamiętaj, że sizeof(void)==1, więc wykonując arytmetykę na wskaźnikach typu void* (np. ptr+2, ptr+3) efektywnie przesuwamy wskaźnik o zadaną liczbę bajtów, a nie elementów w tablicy.

Napisz program, który skorzysta z tej funkcji aby posortować malejąco tablicę 5 losowych liczb zmiennoprzecinkowych (double) z zakresu od 0 do 1, a następnie je wypisze.

Przykład:

```
bsort
0.6108817359077969
0.4425013456210585
0.17523101783136963
0.15975886863052824
0.05746720796712068
```

Wskazówka: losową liczbę z zakresu od 0 do 1 możesz wygenerować jako `rand()/(double)RAND_MAX`.