Zestaw 1

1. Napisz program, który narysuje kwadrat o zadanym boku (liczbie gwiazdek) oraz pusty w środku (też o zadanym boku). O parametry zapytaj w programie, sprawdzając ich poprawność. Przykładowy obraz dla zewnętrznego rozmiaru 10 i wewnętrznego 6.

```
* * * * * * * * * *
* * * * * * * * * *
* *           * *
* *           * *
* *           * *
* *           * *
* *           * *
* *           * *
* *           * *
* *           * *
* * * * * * * * * *
* * * * * * * * * *
```

2. Napisz program, który zapyta o dwa ciągi znaków oraz je porówna, nie zwracając uwagi na wielkość liter (tzn. żeby traktował literę A i a jako takie same). Ciągi znaków można na początku wpisać do odpowiednio pojemnej tablicy char, korzystając z funkcji z języka C (strcpy). Docelowo oczywiście użyjemy std::string, ale tutaj proszę to zrobić „prymitywnie” na bazie wiedzy z języka C. Linię tekstu ze spacjami wczytać można tak:

```
char tablica[1000];
cin.getline(tablica,999);
```

3. Napisz program, który wczytuje zdanie, a potem sprawdza, czy jest palindromem (tzn. czytane wspak brzmi tak samo). Przykład: *Kobyła ma mały bok*. Oczywiście trzeba tu ignorować spacje (oraz nie zwracać uwagi na wielkość liter, proszę je sobie przypomnieć z języka C, np. toupper lub tolower).
4. Napisz funkcję wyliczającą kolejne wyrażenia ciągu Fibonacciego
   a. w wersji rekurencyjnej (czyli funkcja wywołuje samą siebie)
   b. w wersji z jedną pętlą for

   Niech przykładowy program wygląda tak:

```
int main() {
    unsigned long long k = 80;
    for (unsigned long long i=1; i<=k; ++i) {
        cout << fib(i) << endl;
    }
}
```

   Zbadaj jak wygląda czas wykonania tych obliczeń w obu podejściach.

5. Zmodyfikuj program pierwszy tak, żeby korzystał z wyliczonych wcześniej wartości (nie powtarzał ich wyliczania) do wyznaczenia następnych. Na przykład, żeby pytał, który element ciągu ma wyliczyć i żeby program, jeśli już wcześniej wyliczył niższe wartości, to miał je zapamiętane i wykorzystał. Można użyć prostą tablicę lub jeśli ktoś potrafi (chce) to jakiś kontener np. std::vector.