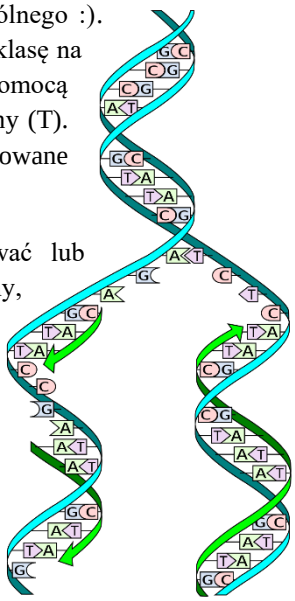
Laboratorium 4 (10pkt)

Zadanie odnosi się do struktury DNA. Jednak z rzeczywistą helisą DNA ma mało wspólnego :). Należy przygotować klasę, która będzie reprezentowała **pojedynczą nić DNA**, oraz drugą klasę na kompletną helisę DNA złożoną z dwóch nici. Informacja w takiej nici zapisana jest za pomocą czterech zasad azotowych (nukleozydów): adeniny (A) i guaniny (G), cytozyny (C) i tyminy (T). W pojedynczej nici nukleozydy połączone są w łańcuch. Natomiast w DNA zasady te skierowane są do wnętrza i tworzą komplementarne pary zasad połączone według schematu: A=T, G=C.



Plik główny jest przygotowany. I należy tylko odpowiednie fragmenty zakomentować lub odkomentować, w zależności od etapu. Plik główny **nie** powinien być modyfikowany, natomiast w plikach nagłówkowych można dokonać pewnych modyfikacji.

Zadbaj o elegancję kodu, unikaj jego powielania, możesz zdefiniować prywatne metody „narzędziowe”.

Etap 1 (3,0 pkt)

Należy przygotować klasę `nicDNA` reprezentującą pojedynczą nić DNA. Nić nukleozydów modelowana jest za pomocą dynamicznej tablicy `nDNA` zawierającej elementy z `enum class nukleozyd`.
W tym etapie należy przygotować:

a) Konstruktor przyjmujący ilość nukleozydów i łańcuch znakowy reprezentujący nukleozydy. Należy go zaimplementować tak, aby działał jako konstruktor bezparametrowy, oraz konstruktor przyjmujący jeden, lub dwa parametry. Bez parametrów ustawia tablicę nukleozydów na **nullptr** i ich ilość na zero. Przy jednym parametrze podawana jest ilość nukleozydów i wszystkie nukleozydy to A.
b) Drugi konstruktor przyjmuje ilość nukleozydów oraz właśnie nukleozyd i ustawia całą nić na ten z argumentu wejściowego.
c) Destruktor.
d) **`operator()`** zwraca znak odpowiadający nukleozydowi na danej pozycji.
e) **`operator<<`** wypisujący nić tak jak na przykładowym wydruku.

Uwaga: Aby uprościć nie trzeba sprawdzać poprawności danych wejściowych, oraz poprawności alokacji pamięci.

Etap 2 (3,0 pkt)

Kolejne elementy klasy `nicDNA`:

a) **`operator[]`** zwraca nukleozyd na odpowiedniej pozycji.
b) **`operator+`** działa jak konkatenacja łańcuchów, łącząc dwie nicie (do pierwszej dołącza drugą) i zwracając trzecią połączoną.
c) Konstruktor kopiujący.
d) Metoda `dolacz`, która dołączy do nici drugą nić podaną w argumencie wejściowym. **Bieżąca nić się powiększa, a dołączana staje się pusta.**

Etap 3 (4.0 pkt)

Etap 3 to klasa `DNA` reprezentująca podwójną nić DNA. Obie pojedyncze nici przechowywane są w polach klasy.

Należy przygotować:

a) Konstruktor bezargumentowy ustawiający puste nici.
b) Konstruktor przyjmujący jedną nić, którą skopiuje do pierwszego pola, a w drugim stworzy drugą nić komplementarną. Czyli tam gdzie w pierwszej jest A, w drugiej powinno być T, gdzie jest G, w drugiej powinno być C i odwrotnie.
c) **`operator()`** zwracający znak odpowiadający nukleozydowi na danej pozycji.
d) **`operator<<`** wypisujący DNA tak jak na przykładowym wydruku.
e) Metoda `klon`, która z jednego DNA stworzy dwa. W argumencie wejściowym mamy „żywność” w postaci DNA. Jeśli w tej „żywności” będzie wystarczająco dużo nukleozydów, aby DNA mogło się podzielić, zwracane jest drugie DNA. Jeśli nie, zwracane DNA jest puste.

Wyniki:

```
====================== ETAP 1 (3.0p)
======================
ndna1:
___
___

ndna2:
AAA

ndna3:
ACAGT
___
|A-
|C-
|A-
|G-
|T-
___

ndna4:
___
|T-
|T-
|T-
|T-
|T-
|T-
|T-
___


====================== ETAP 2 (3.0p)
======================
ndna4:
___
|T-
|C-
|T-
|T-
|T-
|T-
|T-
___

ndna5:
___
|A-
|A-
|A-
|A-
|C-
|A-
|G-
|T-
___

ndna6:
___
|G-
|C-
|T-
___

ndna7:
___
|A-
|C-
|T-
___

ndna6:
___
|G-
|C-
|T-
|A-
|C-
|T-
___

ndna7:
___
___
```

```
====================== ETAP 3 (4.0p)
======================
dna1:
______
______

dna2:
______
|G--C|
|C--G|
|T--A|
|A--T|
|C--G|
|T--A|
______

dna2(1, 3): A

dna4:
______
______

dna6:
______
|G--C|
|C--G|
|T--A|
|A--T|
|C--G|
|T--A|
______
```