Lista 4 – funkcje

1. Zdefiniuj nowe znaczenie dla operatora `*`, tak aby dla dowolnych obiektów `v` oraz `w` klasy `std::vector<double>` wyrażenie `v * w` zwracało iloczyn skalarny `v` i `w`. Przetestuj ten operator (czyli użyj go, np. w funkcji `main`). Wskazówka: chodzi o zwykły iloczyn skalarny, czyli pierwszy wzór na stronie https://en.wikipedia.org/wiki/Dot_product. Możesz założyć, że `v` i `w` mają tę samą liczbę elementów.

2. Zaimplementuj funkcję

   ```
   double root(double (*f)(double), double a, double b, double tolerance = 1e-10)
   ```

   która za pomocą metody bisekcji (https://pl.wikipedia.org/wiki/Metoda_r%C3%B3wnego_podzia%C5%82u) znajduje pierwiastek funkcji `f` w przedziale [`a`, `b`] z dokładnością do `tolerance`. Przetestuj ją, znajdując przy jej pomocy pierwiastki równań

   $\cos(x) = x$ w przedziale [0, 2]

   oraz

   $\cos(x) = 1/2$ w przedziale [0, 1,5],

   oba z tolerancją nieprzekraczającą 10^{-6}. Tolerancja to maksymalne odchylenie otrzymanej wartości od wartości dokładnej.
   Wskazówka: równanie, które rozwiązujesz, zapisuj w postaci f(x) = 0.

3. Napisz program, który będzie wczytywał argumenty wiersza poleceń, a następnie:
   a. jeżeli ich nie ma lub jeżeli wśród nich znajduje się napis `--help` lub `-h`, to wyświetli (dowolny) komunikat o przeznaczeniu programu,
   b. w przeciwnym wypadku będzie traktował swoje argumenty jak liczby kamieni w kolejnych rzędach w grze w Nim (por. https://en.wikipedia.org/wiki/Nim) i wyświetlał optymalny ruch, o ile taki istnieje . Przykład:

   ```
   > nim.exe 1 2 3 4
    z rzędu nr 4 zabierz 4 kamienie
   > nim.exe 1 2 3 4 5 6 7
    rób co chcesz, strategia wygrywająca nie istnieje
   > nim.exe --help
    Program pomaga wygrać w grę nim.
      składnia:
      nim liczba_1 liczba_2 ...
      gdzie liczba_1,...  to liczby kamieni w kolejnych rzędach
   ```

   Wskazówka. Po wykonaniu tego zadania mógłbyś/mogłabyś startować w Olimpiadzie Informatycznej Gimnazjalistów, gdyby ona jeszcze istniała (różna warianty gry w Nim to był jeden z jej ulubionych motywów). Testy możesz przeprowadzić na jednej z licznych stron WWW z implementacją tej gry, np. https://www.transum.org/Software/Nim/. W rozwiązaniu na pewno przyda Ci się operator ^ (czyli XOR, bitowa różnica symetryczna).

Literatura pomocnicza:

- std::vector: https://www.youtube.com/watch?v=PocJ5jXv8No
- function pointers: https://www.youtube.com/watch?v=p4sDgQ-jao4&t=66s
- operators and overloading: https://www.youtube.com/watch?v=mS9755gF66w
- konwersja napisu na liczbę: https://cplusplus.com/reference/string/stoi/
- przykład prostej funkcji `main` z nietrywialnymi argumentami i prasowaniem argumentów wiersza poleceń: https://en.cppreference.com/w/cpp/language/main_function (przykład na końcu strony)
- gra w Nim: https://en.wikipedia.org/wiki/Nim