# Działania na wielomianach

Paweł Wacławik

February 2023

## 1 Problem

Zadanie polegało na zaimplementowaniu czterach działań na wielomianch: dodawanie, odejmowanie, mnożenie i obliczanie wartości wielomianu w punkcie za pomocą algorytmu Hornera. Wielomiany przetrzymuje w tablicy jako kolejne współczynniki stojące przy potęgach.

## 2 Dodawanie wielomianów

Dodawanie wielomianów realizuje jako dodanie mniejszego wielomianu(niższego stopniem) do większego i zwrócenie go:

```
addingPolynomials(first, second) {
    if (first.length >= second.length) {
        return add(first, second);
    } else {
        return add(second, first);
    }

    add(longer, smaller) {
        for (x = 0; x < smaller.length; x++) {
            longer.getPolynomial()[x + longer.getlength - smaller.length] += smaller
            }
        return longer;
    }
```

Dodawanie wielomianów realizuje jako dodawanie mniejszego wielomianu do większego. Złożoność tej operacji wynosi O(n), gdzie n jest stopniem mniejszego wielomianu. Złożoność tyle wynosi, ponieważ muszę przejść przez całą tablicę współczynników wielomianu. Złożoność pamięciowa wynosi O(n), gdzie n to wielkość większego z wielomianów.

# 3 Odejmowanie wielomianów

```
subtractingPolynomials(first, second) {
   longer = 0;
   temp = 0;
   if (first.length >= second.length) {
       longer = first.length;
       temp = first.glength - second.length;
       if (temp < 0) {
           temp *= -1;
       }
   } else {
       longer = second.length;
       temp = second.lLength - first.length;
       if (temp < 0) {
           temp *= -1;
       }
   }
   result = new Polynomial(new double[longer]);
   for (x = 0; x < longer; x++) {
       if (first.length >= second.length) {
           result.getPolynomial()[x] += first.getPolynomial()[x];
           if (x + temp < longer) {
               result.getPolynomial()[x + temp] -= second.getPolynomial()[x];
           }

       } else {
           result.getPolynomial()[x] -= second.getPolynomial()[x];
           if (x + temp < longer) {
               result.getPolynomial()[x + temp] += first.getPolynomial()[x];
           }

       }
   }
   return result;
}
```

Odejmowanie realizuje w następujący sposób: Najpierw sprawdzam, który z wielomianów jest większy tak aby określić rozmiar tablicy dla nowego wielomianu. Następnie tworzę nowy wielomian, który posiada tablicę współczynników o długości równej stopniu większego wielomianu. Następnie wykorzystując to, że nowa tablica współczynników typu double w javie jest wypełniona zerami to: 1. Jeśli wielomian od, którego odejmujemy jest więkzy niż ten drugi to jego wartości dodajemy do końcowego rezultatu, a wartości drugiego wielomianu odejmujemy od końcowego wielomianu tylko, że od odpowiedniego miejsca w tablicy czyli x+(longer.length - smaller.length) tak aby dodawać współczynni-

ki przy tych samych potęgach. Złożoność pamięciowawynosi O(n), gdzie n to wielkość większego z wielomianów.

# 4 Mnożenie wielomianów

```
multiplicationPolynomials(first, second) {
    result = new Polynomial(new double[first.length) + second.glength - 1]);
    for (x = 0; x < first.length; x++) {
        for (y = 0; y < second.length; y++) {
            result.getPolynomial()[x + y] += first.getPolynomial()[x] * second.getPo
        }
    }
    return result;
}
```

Mnożenie realizuje następująco: Tworze nowy wielomian z tablicą wielomianów o rozmaiarze pierwszego + drugiego wielomianu -1. Następnie do następnych komórek wstawiam iloczyn kolejnych współczynników do odpowiednich komórek czyli czyli index pierwszego wielomianu plus index drugiego wielomianu. Złożoność tekiej operacji wynosi $O(n^2)$, ponieważ każdy współczynnik pierwszego wielomianu musi być przemnożony przez każdy współczynnik drugiego wielomianu. Złożoność pamięciowa wynosi O(n + m), gdzie n to wielkość większego z wielomianów i m mniejszego z wielomianów.

# 5 Wartość wielomianu w punkcie(Schemat Hornera)

```
    polynomialValueHorner(polynomial, argument) {
    result = polynomial.getPolynomial()[0];
    for (x = 1; x < polynomial.length; x++) {
        result = result * argument + polynomial.getPolynomial()[x];
    }
    return result;
}
```

Algorytm Hornera jest wykorzystywany do obliczania wartości funkcji w punkcie. Algorytm Hornera działa w następujący sposób:

1.Bierzemy pierwszy współczynnik wielomianu (największa potęga x) i przypisz jego wartość do zmiennej result.

2.Dla każdego kolejnego współczynnika (od mniejszej do mniejszej potęgi x), mnożymy result razy argument i dodaj wartość współczynnika - tak zapisane działanie przypisujemy do result.

3.Zwracam wartość result jako wartość wielomianu dla określonego x.

Złożoność czasowa algorytmu Hornera jest w O(n), gdzie n jest stopniem wielomianu, co oznacza, że jest on dużo szybszy niż tradycyjne obliczanie wielomianu. Złożoność pamięciowawynosi O(n), gdzie n to wielkość wielomianu, w którym liczymy wartość.

# 6 Dokumentacja użytkowa

Aby uruchomoć program należy w terminalu wpisać java -jar program1.jar
Dane wprowadza się w klasie(brak interakcji z użytkownikiem)