Polynomials

Arkadiusz Janus
8 sty 2022

Informacje o projekcie

Projektem który chciałbym tutaj zaprezentować jest implementacja operacji na wielomianach z wykorzystaniem tablic.

W podsekcjach znajduje się szczegółowy opis własności samej implementacji takich jak:

Złożoność

Złożoność poszczególnych operacji możemy podzielić na:

Czasowa

Złożoność czasowa poszczególnych operacji:

- add()
  - **O(N)** - ponieważ musimy przejść przez wszystkie współczynniki magazynowane w tablicy mniejszego wielomianiu
- substract()
  - **O(N)** - ponieważ musimy przejść przez wszystkie współczynniki magazynowane w tablicy mniejszego wielomianiu
- muliply()
  - **O(N*M)** - ponieważ musimy przejść przez wszystkie współczynniki magazynowane w tablicach obu wielomianów
- calculate()
  - **O(N)** - ponieważ musimy przejść przez wszystkie współczynniki wielomianu magazynowane w tablicy
- print()
  - **O(N)** - ponieważ musimy przejść przez wszystkie współczynniki wielomianu magazynowane w tablicy

Gdzie **N** i **M** to liczba współczynników pierwszego i drugiego wielomianu.

Pamięciowa

Złożoność pamięciowa poszczególnych operacji:

- add()
  - **O(N+M)** - ponieważ przechowujemy wszystkie elementy w już wykorzystywanej tablicy większego z wielomianów, ale istnieje druga tablica przechowująca wartości mniejszgo z nich
- substract()
  - **O(N+M)** - ponieważ przechowujemy wszystkie elementy w już wykorzystywanej tablicy większego z wielomianów, ale istnieje druga tablica przechowująca wartości mniejszgo z nich
- muliply()

    - **O(N+M)** - musimy zaalokować nową tablicę będąca sumą rozmiarów obu tablic minus jeden, ale nadal istnieją dwie poprzednie tablice o rozmiarach **N** i **M** , więc całkowita pamięć potrzebna na funkcję wynosi 2(n+m), gdzie dla definicji dużego O, pomijamy stałą
- calculate()
    - **O(1)** - wykorzystujemy do zapisywania wyniku tylko jedną zmienną typu int
- print()
    - Nie wykorzystuje pamięci

Gdzie **N** i **M** to liczba współczynników pierwszego i drugiego wielomianu.

Uruchomienie

Do programu dołączony został plik **MakeFile** dzięki któremu uruchomienie programu sprowadza się do wywołania prostej komendy w wierszu poleceń. Wymaganym jest aby ją wywoływać znajdując się w folderze zawierającym wszystkie niezbędne pliki potrzebne do działania programu.

Nota

Do skorzystania z tej możliwości wymagane jest również posiadanie zainstalowanej aplikacji MakeFile o której informacje można znaleźć na oficjalnej stronie:
https://www.gnu.org/software/make/manual/make.html

Uruchomienie programu

```
make run
```

Uruchomienie testów

```
make test
```

Obsługa

Po uruchomieniu programu zostajemy przywitani przez następujący ekran:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja:
```

Nota

(Po każdej wykonanej operacji będzie on wyświetlany ponownie aż do zakończenia działania programu)

Który pozwala nam skorzystać z następujących funkcjonalności programu:

Dodawanie

Po wybraniu opcji numer 1 i zatwierdzeniu enterem jesteśmy poproszeni o podanie stopnia pierwszego wielomianiu:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 1
Podaj stopien wielomianiu:
```

A następnie jego współczynników przy odpowiednich potęgach x'a:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 1
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
```

Po zebraniu wszystkich elementów program poprosi o stworzenie drugiego wielomianu w dokładnie taki sam sposób.

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 1
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
```

Gdzie po zatwierdzeniu zostaną one utworzone i do siebie dodane co zostanie zasyngnalizowane wypisaniem całego równania wraz ich wynikiem.

```
( 1 + x + x^2 ) + ( 1 + x + x^2 ) = 2 + 2x + 2x^2
```

Odejmowanie

Po wybraniu opcji numer 2 i zatwierdzeniu enterem jesteśmy poproszeni o podanie stopnia pierwszego wielomianiu:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
```

```
5. Wyjdź
Opcja: 2
Podaj stopien wielomianiu:
```

A następnie jego współczynników przy odpowiednich potęgach x'a:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 2
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 2
Podaj wspolczynnik przy x^2: 3
```

Po zebraniu wszystkich elementów program poprosi o stworzenie drugiego wielomianu w dokładnie taki sam sposób.

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 2
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 2
Podaj wspolczynnik przy x^2: 3
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
```

Gdzie po zatwierdzeniu zostaną one utworzone i od siebie odjęte co zostanie zasyngnalizowane wypisaniem całego równania wraz ich wynikiem.

```
( 1 + 2x + 3x^2 ) - ( 1 + x + x^2 ) =  + x + 2x^2
```

Mnożenie

Po wybraniu opcji numer 3 i zatwierdzeniu enterem jesteśmy poproszeni o podanie stopnia pierwszego wielomianiu:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 3
Podaj stopien wielomianiu:
```

A następnie jego współczynników przy odpowiednich potęgach x'a:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
```

```
5. Wyjdź
Opcja: 3
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
```

Po zebraniu wszystkich elementów program poprosi o stworzenie drugiego wielomianu w dokładnie taki sam sposób.

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 3
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
Podaj stopien wielomianiu: 2
Podaj wyraz wolny (x^0): 1
Podaj wspolczynnik przy x: 1
Podaj wspolczynnik przy x^2: 1
```

Gdzie po zatwierdzeniu zostaną one utworzone i przez siebie pomnożone co zostanie zasyngnalizowane wypisaniem całego równania wraz ich wynikiem.

```
( 1 + x + x^2 ) * ( 1 + x + x^2 ) = 1 + 2x + 3x^2 + 2x^3 + x^4
```

Obliczanie wartości

Po wybraniu opcji numer 4 i zatwierdzeniu enterem jesteśmy poproszeni o podanie stopnia wielomianiu który chcemy policzyć:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 4
Podaj stopien wielomianiu:
```

A następnie jego współczynników przy odpowiednich potęgach x'a:

```
Wybierz co chcesz zrobić:
1. Dodaj wielomiany
2. Odejmij wielomiany
3. Pomnóż wielomiany
4. Oblicz wartość wielomianu
5. Wyjdź
Opcja: 4
Podaj stopien wielomianiu: 3
Podaj wyraz wolny (x^0): 2
Podaj wspolczynnik przy x: 3
Podaj wspolczynnik przy x^2: 5
Podaj wspolczynnik przy x^3: -4
Podaj x dla ktorego obliczyc wynik: 3
```

Gdzie po zatwierdzeniu wielomian zostanie utworzony, a wynik dla danej wartości obliczony co zostanie zasyngnalizowane wypisaniem całego wielomianiu i jego wielkości dla podanego argumentu.

```
W(x) = (2 + 3x + 4x^2 - 4x^3)
W(3) = -61
```

Indeks klas

Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Indeks plików

Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

Dokumentacja klas

Dokumentacja klasy Poly

`#include <Poly.h>`

Metody publiczne

- Poly ()
  Podstawowy konstruktor klasy Poly() nie przyjmujący żadnych argumentów.

- Poly (int exponent, int *_coefficients)
  konstruktor klasy Poly() tworzacy obiekt na podstawie podanej w argumentach listy współczynników

- ~Poly ()
  Destruktor obiektu Poly()

- void add (Poly *other)
  Funkcja dodająca jeden wielomian do drugiego.

- void substract (Poly *other)
  Funkcja odejmujaca jeden wielomian od drugiego.

- void multiply (Poly *other)
  Funkcja mnożąca jeden wielomian przez drugi.

- int calculate (int x)
  Funkcja mnożąca jeden wielomian przez drugi.

- void print ()
  Funkcja odpowiadająca za wypisanie wielomianiu w przystępnej formie.

- int * get_coefficients ()
  Funkcja zwracająca listę współczynników (wykorzystywana do testów)

Dokumentacja konstruktora i destruktora

Poly::Poly ()

Podstawowy konstruktor klasy Poly() nie przyjmujący żadnych argumentów.

Poly::Poly (int _*exponent*, int * _*coefficients*)

konstruktor klasy Poly() tworzacy obiekt na podstawie podanej w argumentach listy współczynników

Parametry

_exponent	Stopien wielomianu
coefficients	Lista współczynników wielomianu

Poly::~Poly ()

Destruktor obiektu Poly()

Zwalniane jest zadelarowanie miejsce na listę współczynników

Dokumentacja funkcji składowych

void Poly::add (Poly * *other*)

Funkcja dodająca jeden wielomian do drugiego.

Obliczona wartość nadpisuje obiekt na którym funkcja została wywołana.

Parametry

other	Obiekt typu Poly(), którego wartości zostaną dodane do wartości obiektu

int Poly::calculate (int *x*)

Funkcja mnożąca jeden wielomian przez drugi.

Wykorzystywany jest do tego schemat Hornera, który zaimplementowany algorytm opiera się na wymnożeniu przy każdym kroku poprzedniego wyniku przez podany x i dodanie akutalnie rozpatrywanego współczynnika

Parametry

x	Wartość dla której będzie obliczony wynik

Zwraca

Wynik działania algorytmu

int * Poly::get_coefficients ()

Funkcja zwracająca listę współczynników (wykorzystywana do testów)

Zwraca

Lista współczynników 'coefficients'

void Poly::multiply (Poly * *other*)

Funkcja mnożąca jeden wielomian przez drugi.

Obliczona wartość nadpisuje obiekt na którym funkcja została wywołana.

Parametry

other	Obiekt typu Poly(), którego wartości zostaną pomnożone przez wartości

	obiektu

void Poly::print ()

Funkcja odpowiadająca za wypisanie wielomianiu w przystępnej formie.

void Poly::substract (Poly * *other*)

Funkcja odejmujaca jeden wielomian od drugiego.

Obliczona wartość nadpisuje obiekt na którym funkcja została wywołana.

Parametry

other	Obiekt typu Poly(), którego wartości zostaną odjęte od wartości obiektu

---

Dokumentacja dla tej klasy została wygenerowana z plików:

- Poly.h
- Poly.cpp

Dokumentacja plików

Dokumentacja pliku Info.cpp

Plik stron informacyjnych dokumentacji.

Opis szczegółowy

Plik stron informacyjnych dokumentacji.

Autor

Arkadiusz Janus

Copyright

Dokumentacja pliku main.cpp

Główny plik odpowiadający za interfejs pozwalający użytkownikowi wykorzystać funkcjonalność programu.

```
#include "Poly.h"
#include <iostream>
```

Funkcje

- void menu ()
  Funkcja odpowiedzialna za wyświetlenie menu.

- Poly * create_poly ()
  Funkcja prowadząca użytkownika przez cały proces tworzenia nowego obiektu wielomianiu.

- void add_polys ()
  Funkcja prowadząca użytkownika przez cały proces tworzenia i dodawania dwóch wielomianów do siebie.

- void substract_polys ()
  Funkcja prowadząca użytkownika przez cały proces tworzenia i odejmowania dwóch wielomianów od siebie.

- void multiply_polys ()
  Funkcja prowadząca użytkownika przez cały proces tworzenia i mnożenia dwóch wielomianów przez siebie.

- void calculate_poly ()
  Funkcja prowadząca użytkownika przez cały proces tworzenia obliczania wartości wielomianu dla zadanego argumentu.

- int main ()
  Główna funkcja odpowiadająca za działanie całego programu.

Opis szczegółowy

Główny plik odpowiadający za interfejs pozwalający użytkownikowi wykorzystać funkcjonalność programu.

Autor

Arkadiusz Janus

Dostarczenie samej implementacji wymgała by od użytkownika zapozaniania się z jej szczegółami. Aby maksymalnie uprościć korzystanie z programu, dostarczony wraz z nim jest przejrzysty interfejs który pozwoli każdemu natychmiastowo skorzystać z jego funkcjonalności. Plik zawiera kilka funkcji odpowiedzialnych za obsługę działanie całej implemnetacji.

Copyright

Copyright (c) 2022

Dokumentacja funkcji

void add_polys ()

Funkcja prowadząca użytkownika przez cały proces tworzenia i dodawania dwóch wielomianów do siebie.

Bazuje na funkcji create_poly(), które tworzy wielomiany, a następnie na pierwszym z nich jest wywoływana metoda add(), z drugim z nich jako argument.

void calculate_poly ()

Funkcja prowadząca użytkownika przez cały proces tworzenia obliczania wartości wielomianu dla zadanego argumentu.

Bazuje na funkcji create_poly(), które tworzy wielomian, a następnie prosi użytkownika o podanie argumentu dla niego według której będzie obliczana wartość. Mając wszystkie dane wywoływana jest na obiekcie metoda calculate(), która zwraca wynik.

Poly * create_poly ()

Funkcja prowadząca użytkownika przez cały proces tworzenia nowego obiektu wielomianiu.

Zwraca

Poly*

int main ()

Główna funkcja odpowiadająca za działanie całego programu.

Za pomocą pętli pozwala na nieskończone działanie pozwalające na wykonywanie potrzebynch nam operacji bez konieczności uruchamiania programu za każdym razem. Z każdą iteracją wypisuje menu za pomocą funkcji menu() i prosi użytkownika o wybranie funkcjonalności z której chciałby skorzystać.

Zwraca

int

void menu ()

Funkcja odpowiedzialna za wyświetlenie menu.

void multiply_polys ()

Funkcja prowadząca użytkownika przez cały proces tworzenia i mnożenia dwóch wielomianów przez siebie.

Bazuje na funkcji create_poly(), które tworzy wielomiany, a następnie na pierwszym z nich jest wywoływana metoda multiply(), z drugim z nich jako argument.

void substract_polys ()

Funkcja prowadząca użytkownika przez cały proces tworzenia i odejmowania dwóch wielomianów od siebie.

Bazuje na funkcji create_poly(), które tworzy wielomiany, a następnie na pierwszym z nich jest wywoływana metoda substract(), z drugim z nich jako argument.

Dokumentacja pliku Poly.cpp

Plik implementujący funkcjonalność obiketu Poly()

```
#include "Poly.h"
```

Opis szczegółowy

Plik implementujący funkcjonalność obiketu Poly()

Autor

Arkadiusz Janus

Copyright

Dokumentacja pliku Poly.h

Plik nagłówkowy odpowiadający za implementację wielomianów.
`#include <iostream>`

Komponenty

- class Poly

Opis szczegółowy

Plik nagłówkowy odpowiadający za implementację wielomianów.

Autor

Arkadiusz Janus

Wielomiany są zaimplementowane za pomocą tablicy w której są przechowywane jego współczynniki. Każde miejsce odpowiada odpowiedniemu współczynnikowi dla danej potęgi wielomianu. Ilośc potęg jest wyznaczana na podstawie stopnia wielomianu który jest podawany przy tworzeniu.

Klasa zawiera dwa konstruktory i destruktor, a także funkcje które pozwalają wykonywać operacje na wielomianach:

- dodawanie
- odejmowanie
- mnożenie
- oblicznie dla danej wartości

Copyright

Poly.h

```
Idź do dokumentacji tego pliku.1 #include<iostream>
2
22 class Poly{
23     private:
24         int *coefficients;
25         int exponent;
26
27
28     public:
29         Poly();
30         Poly(int exponent, int* _coefficients);
31         ~Poly();
32         void add(Poly* other);
33         void substract(Poly* other);
34         void multiply(Poly* other);
35         int calculate(int x);
36         void print();
37         int* get coefficients();
38 };
```

Dokumentacja pliku test_poly.cpp

Plik odpowiadający za testy wszystkich funkcjonalności klasy Poly()

```
#include "Poly.h"
#include <iostream>
```

Funkcje

- void test_print ()
  Funkcja testujaca wypisywanie wieomianów dla różnych przypadków.

- bool compare_arrays (int *array1, int *array2, int size)
  Funkcja typu assert porównująca dwie tablice.

- void test_add ()
- void test_substract ()
- void test_multiply ()
- void test_calculate ()
- int main ()

Opis szczegółowy

Plik odpowiadający za testy wszystkich funkcjonalności klasy Poly()

Autor

Arkadiusz Janus

Do testowania funkcjonalności podszedłem w następujący sposób. Dla każdej z nich zaimplementowałem te same operacje które znajdują się w klasie, lecz operując tylko na tablicach i następnie porównując wyniki za pomocą funkcji pomocniczej compare_arrays(int*, int*, int) i porównań.

Copyright

Copyright (c) 2022

Dokumentacja funkcji

bool compare_arrays (int * *array1*, int * *array2*, int *size*)

Funkcja typu assert porównująca dwie tablice.

Parametry

in	*array1*	pierwsza tablica do porównania
in	*array2*	druga tablica do porównania
in	*size*	wielkość obu tablic (nie mogą być różne)

int main ()

void test_add ()

Funkcja testująca dodawanie wielomianów dla przypadków:

- n = m
- n > m
- n < m
- n = m = 0 gdzie n i m to stopnie wielomianów

void test_calculate ()

Funkcja testująca obliczanie wartości wielomianu dla zadanego argumentu.

void test_multiply ()

Funkcja testująca mnożenie wielomianów dla przypadków:

- n = m
- n > m
- n < m
- n = m = 0 gdzie n i m to stopnie wielomianów

void test_print ()

Funkcja testujaca wypisywanie wieomianów dla różnych przypadków.

void test_substract ()

Funkcja testująca odejmowanie wielomianów dla przypadków:

- n = m
- n > m
- n < m
- n = m = 0 gdzie n i m to stopnie wielomian