7. Obliczanie wartości wielomianu

Pojęcie wielomianu znasz z lekcji matematyki. Wielomianów używa się m.in. do opisu zależności między wielkościami fizycznymi w modelowaniu różnych zjawisk. Mają zastosowanie także w informatyce. Za ich pomocą można wyznaczyć przybliżoną wartość funkcji, gdy dokładna wartość jest trudna do znalezienia (np. w przypadku funkcji trygonometrycznej). W tym temacie zajmiemy się obliczaniem wartości wielomianu. Wykorzystamy do tego różne algorytmy: algorytm naiwny oraz algorytm, który wykona mniej działań arytmetycznych.

Cele lekcji

- Obliczysz wartość wielomianu algorytmem naiwnym.
- Wyznaczysz wartość wielomianu algorytmem optymalnym – wykorzystującym schemat Hornera.
- Dowiesz się, w których z omawianych do tej pory problemów użyliśmy schematu Hornera.

7.1. Sformułowanie problemu obliczania wartości wielomianu

Wielomian Przypomnijmy, że **wielomianem** stopnia n zmiennej rzeczywistej x nazywamy funkcję postaci:

$$W(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_2 x^2 + a_1 x + a_0$$

gdzie $a_n \neq 0$, n – liczba naturalna dodatnia, $a_n, a_{n-1}, \ldots, a_2, a_1, a_0$ – współczynniki wielomianu będące liczbami rzeczywistymi. Współczynnik a_0 nazywamy wyrazem wolnym.

W temacie nie będziemy się zajmować wielomianem zerowym, czyli funkcją $W(x) = 0$. Oto specyfikacja problemu obliczania wartości wielomianu dla danego argumentu:

Specyfikacja

Dane: n – liczba całkowita dodatnia oznaczająca stopień wielomianu,
A[0..n] – tablica liczb rzeczywistych będących współczynnikami wielomianu, A[n] ≠ 0, element A[0] odpowiada współczynnikowi a_0, A[1] – współczynnikowi a_1 itd.,
x – liczba rzeczywista.

Wynik: y – wartość wielomianu o współczynnikach z tablicy A dla argumentu x.

7.2. Obliczanie wartości wielomianu algorytmem naiwnym

Wartość wielomianu dla danego argumentu można policzyć, podstawiając ten argument bezpośrednio do wzoru wielomianu. Taki algorytm jest algorytmem naiwnym. Sumowanie wyrazów wielomianu warto rozpocząć od wyrazu wolnego i wykorzystywać policzoną aktualnie potęgę x do wyliczenia kolejnej ($x^n = x^{n-1} \cdot x$ dla $n > 1$). Oto zapis algorytmu w pseudokodzie:

```
y ← A[0]
z ← 1
dla i ← 1, 2, …, n wykonuj
    z ← z * x
    y ← y + A[i] * z
```

Wartością początkową wyniku (zmiennej y) jest wartość wyrazu wolnego (A[0]). Kolejne potęgi wartości zmiennej x będziemy pamiętać w zmiennej pomocniczej z. Na początku przyjmuje ona wartość 1 (czyli x^0). Na potrzeby naszych obliczeń przyjmiemy, że $0^0 = 1$.

Chcemy obliczać wartości wielomianów różnego stopnia, a więc liczba współczynników będzie się zmieniać. Dlatego współczynniki zapamiętamy w zmiennej typu **vector**.

Kod źródłowy funkcji Czytaj, wczytującej stopień wielomianu oraz jego współczynniki, może wyglądać następująco:

Kod źródłowy funkcji wczytującej stopień oraz współczynniki wielomianu

```
1. void Czytaj(vector<float> &A)
2. {
3.     int n;
4.     cout<<"Stopien wielomianu: ";
5.     cin>>n;
6.     A.resize(n+1);
7.     for (int i=n;i>=0;i--)
8.     {
9.         cout<<"a"<<i<<" = ";
10.        cin>>A[i];
11.    }
12. }
```

Parametr A typu **vector**, pamiętający współczynniki wielomianu, przekazywany jest przez referencję (linia 1), ponieważ funkcja określa jego wartość. W linii 5 wczytywany jest stopień wielomianu. Następnie w linii 6 rozmiar parametru A jest ustalany na n+1 (ponieważ parametr A przechowuje współczynniki od a_0 do a_n). Pętla w liniach 7–11 wczytuje współczynniki wielomianu, zaczynając od współczynnika a_n przy najwyższej potędze x (element A[n]).

Kod źródłowy funkcji realizującej algorytm naiwny obliczania wartości wielomianu może być następujący.

Dobra rada
Algorytm naiwny to algorytm wykorzystujący najbardziej intuicyjną metodę. Pamiętaj, że takie algorytmy są często mało efektywne.

Dobra rada
Pamiętaj, że korzystanie z typu **vector** jest możliwe po dołączeniu do programu biblioteki vector (dyrektywa #include <vector>).

Dobra rada
Nie musisz przekazywać na zewnątrz funkcji stopnia wielomianu (rozmiaru parametru typu **vector**), ponieważ jest on dostępny dzięki metodzie **size**.

Kod źródłowy funkcji obliczającej wartość wielomianu algorytmem naiwnym

```
1. float W(vector<float> A, float x)
2. {
3.     float y=A[0], z=1;
4.     for (int i=1;i<A.size();i++)
5.     {
6.         z=z*x;
7.         y=y+A[i]*z;
8.     }
9.     return y;
10. }
```

W linii 3 deklarowane są dwie zmienne y i z: pierwszą z nich wykorzystamy do sumowania wyrazów wielomianu, drugą – do obliczania potęg x. Zmiennym y i z przy deklaracji nadawane są wartości początkowe. Zwróć uwagę na zakres zmiennej i, sterującej pętlą **for** (linia 4). Zmienia się on od 1 (jako pierwszy w pętli będzie wykorzystany współczynnik a_1) do wartości A.**size**()-1 (rozmiar zmiennej typu **vector** wynosi n+1, więc współczynnik a_n ma indeks A.**size**()-1). W linii 6 wartość zmiennej z mnożona jest przez wartość zmiennej x, wskutek czego otrzymujemy wartość x^i. W linii 7 do wartości zmiennej y dodawany jest i-ty wyraz wielomianu.

Rysunek 7.1 przedstawia efekt wywołania programu dla wielomianu $x^3 + 2x^2 + 3x + 4$ i argumentu $x = 2$.

```
Stopien wielomianu: 3
a3 = 1
a2 = 2
a1 = 3
a0 = 4
x = 2
W(2) = 26
```

Rys. 7.1. Efekt wywołania programu dla wielomianu $x^3 + 2x^2 + 3x + 4$ oraz argumentu $x = 2$

Ćwiczenie 1

Napisz program, który obliczy wartość wielomianu algorytmem naiwnym zgodnie ze specyfikacją podaną na s. 122. Wykorzystaj przedstawione w temacie funkcje `Czytaj` i `W`.

Omówiony algorytm naiwny obliczania wartości wielomianu wykonuje $2n$ operacji mnożenia oraz n operacji dodawania. Warto się zastanowić, czy nie można przekształcić wzoru ogólnego funkcji wielomianowej, aby wykonywać mniej działań arytmetycznych.

7.3. Obliczanie wartości wielomianu za pomocą schematu Hornera

Liczbę operacji arytmetycznych podczas wyznaczania wartości wielomianu możemy zmniejszyć, jeśli wykorzystamy schemat Hornera. Jego działanie wyjaśnimy na przykładzie wielomianu stopnia 3. Z trzech pierwszych wyrazów wyłączmy x przed nawias.

$$a_3 \cdot x^3 + a_2 \cdot x^2 + a_1 \cdot x + a_0 = x \cdot (a_3 \cdot x^2 + a_2 \cdot x + a_1) + a_0$$

W nawiasie otrzymujemy wielomian stopnia o 1 mniejszego. Dla tego wielomianu (w tym przypadku trójmianu kwadratowego) postępujemy w ten sam sposób – wyłączamy x przed nawias.

$$x \cdot (a_3 \cdot x^2 + a_2 \cdot x + a_1) + a_0 = x \cdot (x \cdot (a_3 \cdot x + a_2) + a_1) + a_0$$

Obliczając wartość wielomianu z tego wzoru, wykonamy 3 mnożenia i 3 dodawania. Gdybyśmy skorzystali ze wzoru funkcji wprost, wykonalibyśmy także 3 dodawania, ale 6 operacji mnożenia.

Podobnie będziemy postępować dla wielomianu stopnia n: wielokrotnie wyłączamy x przed nawias, aż do otrzymania w najbardziej wewnętrznych nawiasach dwumianu $a_n \cdot x + a_{n-1}$. Dla wielomianu stopnia n wykonamy n operacji mnożenia (zamiast $2n$). Jest to minimalna liczba operacji mnożenia, jaką trzeba wykonać, aby obliczyć wartość wielomianu. Algorytm wykorzystujący ten wzór jest więc algorytmem optymalnym.

Dobra rada
Pamiętaj, że algorytm optymalny to algorytm wykonujący najmniejszą możliwą liczbę operacji charakterystycznych dla danego problemu. Nie istnieje rozwiązanie tego problemu, które wykorzystuje mniejszą liczbę operacji.

Dla wielomianu stopnia n **schemat Hornera** można zapisać wzorem:

Schemat Hornera

$$W(x) = x \cdot (x \cdot \ldots \cdot (a_n \cdot x + a_{n-1}) + \ldots + a_1) + a_0$$

Algorytm obliczający wartość wielomianu z wykorzystaniem schematu Hornera można zapisać następująco:

```
y ← A[n]
dla i ← n – 1, n – 2, …, 0 wykonuj
    y ← x * y + A[i]
```

Obliczenia rozpoczynamy od współczynnika przy najwyższej potędze (wartość początkowa zmiennej y). W pętli mnożymy dotychczasową wartość zmiennej y przez x i dodajemy kolejny współczynnik. Kod źródłowy funkcji realizującej ten algorytm znajduje się na s. 126.

A to ciekawe

Horner i początki animacji

Nazwa „schemat Hornera" pochodzi od nazwiska angielskiego matematyka Williama G. Hornera (1786–1837). Znany jest on nie tylko z dokonań matematycznych – interesował się też m.in. optyką. W 1834 r. skonstruował zootrop (ang. *zoetrope*) – urządzenie służące do uzyskiwania ruchomego obrazu. Ma ono postać obrotowego bębna, wewnątrz którego znajduje się papierowa taśma z sekwencją rysunków przedstawiających kolejne etapy ruchu. Po wprawieniu bębna w ruch przez pionowe szczeliny można zaobserwować ruchomy obraz. Zootrop upowszechnił się jako zabawka w latach 60. XIX w.

Kod źródłowy funkcji obliczającej wartość wielomianu z wykorzystaniem schematu Hornera

```
float W(vector<float> A, float x)
{
    int n=A.size()-1;
    float y=A[n];
    for (int i=n-1;i>=0;i--) y=x*y+A[i];
    return y;
}
```

Kod źródłowy funkcji wczytującej stopień oraz współczynniki wielomianu, s. 123

Funkcja wczytująca stopień wielomianu oraz jego współczynniki będzie wyglądała tak samo jak w przypadku programu wyznaczającego wartość wielomianu algorytmem naiwnym.

Ćwiczenie 2

Napisz program, który obliczy wartość wielomianu z wykorzystaniem schematu Hornera, zgodnie ze specyfikacją podaną na s. 122.

Zapamiętaj

Stosując schemat Hornera, wielokrotnie wyłącza się argument przed nawias we wzorze ogólnym funkcji wielomianowej, aż w najbardziej wewnętrznych nawiasach otrzymamy wielomian pierwszego stopnia. Z powstałego wzoru można obliczyć wartość wielomianu stopnia *n*, wykonując *n* operacji mnożenia i *n* operacji dodawania. Jest to algorytm optymalny.

7.4. Inne zastosowania schematu Hornera

Algorytmy zamiany liczby dwójkowej na dziesiętną, podręcznik *Informatyka na czasie 2. Zakres rozszerzony*, s. 50–52

Zamiana liczby w systemie pozycyjnym o wybranej podstawie na liczbę dziesiętną, podręcznik *Informatyka na czasie 2. Zakres rozszerzony*, s. 55–56

Schemat Hornera wykorzystywaliśmy podczas **zamiany liczby dwójkowej na dziesiętną** oraz podczas **zamiany liczby w systemie pozycyjnym o wybranej podstawie na liczbę dziesiętną**. W tych algorytmach rolę współczynników wielomianu pełnią cyfry reprezentacji liczby w systemie o podstawie p. Stopień wielomianu określa liczba cyfr (stopień jest o jeden mniejszy od liczby cyfr), a podstawa systemu jest argumentem, dla którego obliczamy wartość wielomianu.

Wartość dziesiętną liczby w systemie pozycyjnym o podstawie p zapisanej za pomocą n cyfr, gdzie c_i oznacza i-tą cyfrę liczby przeglądanej od prawej do lewej, $0 \leqslant i < n$, obliczymy ze wzoru:

$$c_{n-1} \cdot p^{n-1} + c_{n-2} \cdot p^{n-2} + \ldots + c_1 \cdot p + c_0$$

Podany wzór to sposób obliczania wartości wielomianu o współczynnikach $c_{n-1}, \ldots, c_0$ dla argumentu p.

Rozważamy wielomian $W(x)$ stopnia $n - 1$:

$$W(x) = c_{n-1} \cdot x^{n-1} + \ldots + c_2 \cdot x^2 + c_1 \cdot x + c_0$$

Obliczmy wartość $W(p)$, korzystając ze schematu Hornera:

$$W(p) = p \cdot \left(p \cdot \ldots \cdot (c_{n-1} \cdot p + c_{n-2}) + \ldots + c_1\right) + c_0$$

Tego sposobu używaliśmy do obliczania wartości dziesiętnej liczby zapisanej w systemie pozycyjnym o podstawie innej niż 10.

Algorytm szybkiego podnoszenia do potęgi, podręcznik *Informatyka na czasie 2. Zakres rozszerzony*, s. 57–58

Schemat Hornera stosuje się także w algorytmie obliczającym potęgę liczby. W wersji iteracyjnej **algorytmu szybkiego podnoszenia do potęgi** wykorzystywaliśmy zapis wykładnika potęgi w systemie binarnym. Dokładniej – przeglądaliśmy cyfry binarne wykładnika w kolejności od prawej do lewej, a więc w kolejności, w jakiej są one uzyskiwane podczas zamiany liczby dziesiętnej na binarną.

Jeśli znamy zapis wykładnika w postaci binarnej, możemy także zapisać algorytm szybkiego podnoszenia do potęgi, który przegląda cyfry binarne wykładnika od lewej do prawej, czyli od cyfry najbardziej znaczącej. Korzystamy wtedy wprost ze schematu Hornera. Na przykład:

$$x^{13} = x^{1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0} = x^{2 \cdot (1 \cdot 2^2 + 1 \cdot 2 + 0) + 1} = x^{2 \cdot (2 \cdot (1 \cdot 2 + 1) + 0) + 1} =$$
$$= (x^{2 \cdot (1 \cdot 2 + 1) + 0})^2 \cdot x = \left((x^{1 \cdot 2 + 1})^2\right)^2 \cdot x = \left((x^2 \cdot x)^2\right)^2 \cdot x$$

Algorytm wykorzystujący ten wzór można zapisać w pseudokodzie tak jak poniżej. Zakładamy, że napis s jest reprezentacją binarną wykładnika potęgi. Wartość zmiennej y jest policzoną potęgą liczby x.

```
y ← 1
dla i ← 0, 1, …, długość(s) - 1 wykonuj
    y ← y * y
    jeśli s[i]='1' to y ← y * x
```

Dobra rada

Jeśli wykładnik potęgi jest reprezentowany w systemie dziesiętnym, zastosuj algorytm przeglądający cyfry binarne od prawej do lewej, ponieważ wówczas nie musisz pamiętać reprezentacji binarnej wykładnika potęgi.

W każdym kroku algorytmu podnosimy aktualną wartość wyniku do kwadratu (wartością początkową jest 1, czyli x^0). Jeśli w zapisie binarnym wykładnika występuje jedynka, to dodatkowo mnożymy wynik przez podstawę potęgi.

Rekurencyjny algorytm szybkiego podnoszenia do potęgi, podręcznik *Informatyka na czasie 2. Zakres rozszerzony*, s. 241–244

Zwróć uwagę, że przedstawiony algorytm jest iteracyjną wersją **rekurencyjnego algorytmu szybkiego podnoszenia do potęgi**. W rekurencyjnej wersji algorytmu reprezentacja binarna wykładnika była pamiętana niejawnie na stosie.

Ćwiczenie 3

Napisz program, który wczyta podstawę potęgi oraz napis reprezentujący wykładnik potęgi w systemie binarnym, a następnie wypisze wartość potęgi. Zastosuj algorytm szybkiego podnoszenia do potęgi z wykorzystaniem schematu Hornera.

Zapamiętaj

Schemat Hornera pozwala ograniczać liczbę operacji wykonywanych przez algorytmy. Wykorzystuje się go np. w algorytmach zamiany reprezentacji liczby w systemie pozycyjnym o podstawie innej niż 10 na reprezentację w systemie dziesiętnym, a także w algorytmie szybkiego podnoszenia do potęgi.

Podsumowanie

- Wartość wielomianu można obliczyć algorytmem naiwnym, który wykonuje $2n$ operacji mnożenia i n operacji dodawania, gdzie n oznacza stopień wielomianu.
- Schemat Hornera polega na wielokrotnym wyłączaniu argumentu przed nawias we wzorze ogólnym funkcji wielomianowej, aż w najbardziej wewnętrznych nawiasach otrzymamy wielomian pierwszego stopnia.
- Obliczając wartość wielomianu stopnia n z wykorzystaniem schematu Hornera, wykonujemy n operacji mnożenia i n operacji dodawania. Algorytm ten jest optymalny.
- Schemat Hornera stosuje się m.in. do obliczania wartości dziesiętnej liczb zapisanych w systemie o innej podstawie oraz do szybkiego podnoszenia do potęgi.

Zadania

* **1** Napisz program, który wczyta z klawiatury dwa wielomiany, a następnie wyznaczy wielomian będący ich sumą i go wypisze.

* **2** Napisz program, który z pliku tekstowego przekazanego ci przez nauczyciela (np. *wielomian.txt*) wczyta stopień i współczynniki wielomianu, a następnie dla argumentu wczytanego z klawiatury obliczy wartość wielomianu, korzystając ze schematu Hornera. Stopień wielomianu znajduje się w pierwszym wierszu pliku tekstowego, w kolejnych wierszach podane są współczynniki wielomianu.

* **3** Napisz program, który wczyta z klawiatury podstawę systemu pozycyjnego z zakresu od 2 do 9, liczbę cyfr liczby oraz cyfry tej liczby i obliczy jej wartość dziesiętną, korzystając ze schematu Hornera.

** **4** Napisz program obliczający wartość wielomianu zgodnie ze specyfikacją przedstawioną na s. 122, z wykorzystaniem schematu Hornera. Funkcję realizującą schemat Hornera zapisz rekurencyjnie.

** **5** Napisz program, który wczyta z klawiatury dwa wielomiany, a następnie wyznaczy wielomian będący ich iloczynem i go wypisze.
Wskazówka: Stopień wielomianu będącego iloczynem wielomianów to suma stopni wielomianów składowych.

*** **6** Napisz program, który wczyta z klawiatury wielomian oraz liczbę całkowitą c i obliczy iloraz wielomianu przez dwumian $x - c$. Zastosuj schemat Hornera.

*** **7** Napisz program obliczający przybliżoną wartość $\sin x$ ze wzoru:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \dots$$

dla $0 \leqslant x \leqslant 6{,}28$. Obliczenia zakończ po zsumowaniu sześciu pierwszych wyrazów. Wykorzystaj schemat Hornera, a wartości silni zapamiętaj w tablicy.

8. Metody obliczeń przybliżonych

W wielu sytuacjach wymagających zastosowania obliczeń łatwo jest otrzymać dokładny wynik – np. w wyborach parlamentarnych, w których zliczany jest każdy oddany głos. Istnieją jednak problemy, których nie potrafimy rozwiązać za pomocą znanych algorytmów znajdujących dokładne wyniki lub algorytmy takie są czasochłonne. Można wówczas wykorzystać metody, które dają wyniki przybliżone.

Cele lekcji

- Znajdziesz miejsce zerowe funkcji metodą bisekcji.
- Obliczysz pierwiastek kwadratowy z liczby metodą bisekcji i metodą Newtona–Raphsona.
- Poznasz metodę prostokątów i metodę trapezów, wykorzystywane do obliczania pól obszarów zamkniętych.
- Zastosujesz metodę Monte Carlo do policzenia przybliżonej wartości liczby π oraz do symulacji ruchów Browna.

W poprzednich tematach zajmowaliśmy się m.in. **znajdowaniem pierwiastków równania kwadratowego** oraz **obliczaniem wartości wielomianu**. Choć przedstawione tam algorytmy umożliwiają znalezienie dokładnych wyników, to programy realizujące te algorytmy podają rozwiązania obarczone błędami. Wynikają one jednak z reprezentacji liczb w komputerze, a nie z samych algorytmów. Teraz zajmiemy się problemami i metodami ich rozwiązania, w których przybliżony wynik jest przede wszystkim skutkiem zastosowanego algorytmu.

Algorytmy znajdowania pierwiastków równania kwadratowego, s. 115–119

Obliczanie wartości wielomianu, s. 122–126

8.1. Znajdowanie miejsc zerowych funkcji

Jedną z metod znajdowania miejsc zerowych funkcji, czyli argumentów, dla których funkcja przyjmuje wartość 0, jest **metoda bisekcji (połowienia)**. Postępuje się w niej podobnie jak w **algorytmie przeszukiwania binarnego** – w kolejnych krokach zmniejszamy o połowę zakres przeszukiwanych danych.

W ogólnym przypadku w metodzie bisekcji konstruujemy ciąg, którego wyrazami są przybliżenia poszukiwanej wartości, a granicą tego ciągu jest poszukiwana wartość. Kolejne wyrazy ciągu to środki coraz mniejszych przedziałów, które przeszukujemy. Zakres poszukiwań zawężamy o połowę, zastępując jeden z końców przedziału środkiem przedziału. Kończymy, gdy wartość kolejnego wyrazu ciągu jest równa szukanej wartości lub gdy długość przeszukiwanego przedziału jest nie większa od przyjętej dokładności obliczeń (wynik jest wówczas przybliżony – jest nim ostatnio wyznaczony środek przedziału).

Metoda bisekcji (połowienia)

Przeszukiwanie binarne, podręcznik *Informatyka na czasie 2. Zakres rozszerzony*, s. 154