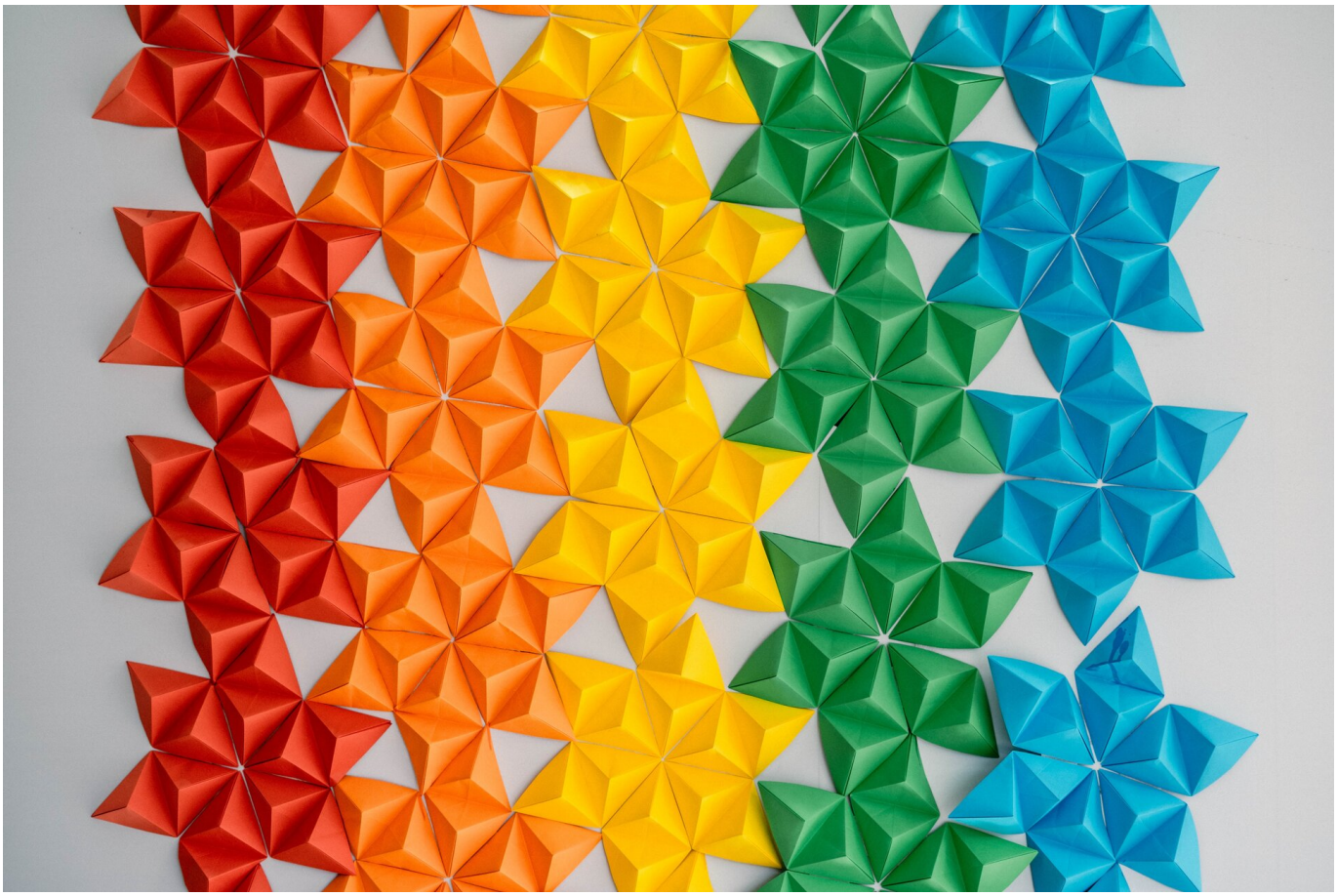

Rekurencja a iteracja

Źródło: Chinh Le Duc, domena publiczna.

Niektóre iteracje mogą być zapisane jako rekurencje, a rekurencje zwykle możemy zastąpić iteracjami… Czy jednak rozumiesz różnicę między dwiema metodami? Zarówno podejście rekurencyjne, jak i iteracyjne oparte jest na powtórzeniach. Jakie są zatem różnice?
W przypadku iteracji wielokrotnie wykonywany jest ciąg instrukcji w pętli. Jeżeli chodzi o rekurencję, funkcja wywołuje samą siebie, w związku z czym dane są często głęboko zagnieżdżane.

O tym, jak zagadnienie rekurencji wyjaśnia matematyka, przeczytasz w e-materiałach:

- Ciąg określony rekurencyjnie,
- Ciąg geometryczny określony rekurencyjnie,
- Wzór ogólny ciągu określonego rekurencyjnie,
- Ciąg arytmetyczny określony wzorem rekurencyjnym.

Porównanie implementacji algorytmów metodą rekurencyjną i iteracyjną przedstawiamy w e-materiałach:

- Rekurencja a iteracja w języku C++,
- Rekurencja a iteracja w języku Java,
- Rekurencja a iteracja w języku Python.

Więcej zadań? Sięgnij do: Rekurencja a iteracja w zadaniach.

> **Twoje cele**
>
> - Podsumujesz podstawowe informacje o iteracji i rekurencji.
> - Wyjaśnisz, czy można zastąpić rekurencję iteracją i odwrotnie.
> - Przeanalizujesz i porównasz algorytmy rekurencyjny i iteracyjny obliczania wartości elementów ciągu.

Przeczytaj

Iteracja i rekurencja – przypomnienie

Przypomnijmy sobie podstawowe informacje na temat iteracji i rekurencji.

Iteracja:

- polega na wielokrotnym wykonywaniu zapisanych w programie instrukcji,
- jest realizowana za pomocą pętli – instrukcji iteracyjnych,
- musi być wprowadzony warunek zakończenia pętli.

Rekurencja:

- funkcja wywołuje samą siebie w celu osiągnięcia rozwiązania problemu,
- musi spełniać dwa warunki:
    - warunek początkowy – warunek, który spełniony nie spowoduje wywołania funkcji rekurencyjnej, tylko zwróci konkretną wartość,
    - co najmniej jeden argument w rekurencyjnym wywołaniu funkcji musi mieć inną wartość niż poprzednie.

Którą metodę wybrać?

Dowolny problem możemy rozwiązać dwiema technikami – rekurencyjną i iteracyjną. Problem obliczania wartości elementu ciągu rekurencyjnego możemy także rozwiązać w sposób iteracyjny.

Warto jednak zastanowić się, której metody użyć w algorytmie podczas rozwiązywania danego problemu. Stosowanie techniki rekurencyjnej czyni kod często czytelniejszym, krótszym i zrozumiałym. W przypadku iteracji nasz kod może być dłuższy i trudniejszy do zrozumienia. Trzeba wziąć pod uwagę, że rozwiązując niektóre problemy metodą rekurencyjną, płacimy wysoką cenę. Często funkcja rekurencyjna jest wywoływana dużą liczbę razy, a obliczenia są nieefektywne. W tych przypadkach zdecydowanie wybieramy metodę iteracyjną.

Obliczanie wartości elementu ciągu

Sprawdźmy, jak – na podstawie kilku prostych ćwiczeń – obliczyć wartości elementu ciągu. Przy rozwiązaniu zadań skorzystamy zarówno z techniki rekurencyjnej, jak i z iteracyjnej.

> **Problem 1**
>
> Zdefiniowany jest następujący ciąg:
>
> $$a_n = \begin{cases} 1 & gdy\ n = 0 \\ 4 \cdot a_{n-1} + 2 & gdy\ n > 0 \end{cases}$$
>
> Napisz, za pomocą ciągu instrukcji pseudokodu, funkcję której argumentem będzie numer szukanego wyrazu ciągu i która zwróci wartość tego wyrazu.

Przeanalizujmy zapisaną w postaci pseudokodu funkcję realizującą algorytm obliczania wartości n-tego elementu zdefiniowanego ciągu, przy użyciu **techniki rekurencyjnej**.

```
funkcja rekurencyjnie(n)
    jeżeli n == 0 wykonaj:
        zwróć 1
    w przeciwnym razie wykonaj:
        zwróć 4 * rekurencyjnie(n-1) + 2
```

Argumentem funkcji `rekurencyjnie(n)` jest numer wyrazu ciągu, który chcemy obliczyć. W przypadku zerowego wyrazu ciągu zwracana jest wartość 1, natomiast w innych wypadkach musi zostać obliczona wartość poprzedniego wyrazu ciągu, poprzez wywołanie odpowiedniej funkcji rekurencyjnej z argumentem zmniejszonym o 1.

Problem ten możemy także rozwiązać iteracyjnie. Przeanalizujmy zapisaną za pomocą pseudokodu funkcję, tym razem w **wersji iteracyjnej**.

```
funkcja iteracyjnie(n)
    a = 1
    dla i = 1, 2, ..., n wykonuj:
        a = 4 * a + 2
    zwróć a
```

Jak widać, powyższa funkcja jest również zrozumiała i krótka. Zmiennej `a` przypisywana jest wartość 1, to znaczy wartość naszego zerowego elementu ciągu. Następnie dla kolejnych elementów obliczane są wartości w pętli, aż do osiągnięcia naszego poszukiwanego wyrazu o numerze `n`. Wtedy zwracana jest jego wartość `a`.

Przejdźmy teraz do trudniejszego przykładu, w którym wartość elementu ciągu będzie zależała od dwóch poprzednich wyrazów ciągu:

> **Problem 2**
>
> Zdefiniowany jest następujący ciąg:
>
> $$a_n = \begin{cases} 0 & \text{gdy } n = 0 \\ 1 & \text{gdy } n = 1 \\ a_{n-1} \cdot a_{n-2} + 3 & \text{gdy } n > 1 \end{cases}$$
>
> Napisz, wykorzystując ciąg instrukcji pseudokodu, funkcję, której argumentem będzie numer szukanego wyrazu ciągu i która zwróci wartość tego wyrazu.

Zacznijmy od **metody rekurencyjnej**:

```
funkcja rekurencyjnieDrugie(n)
    jeżeli n == 0 wykonaj:
        zwróć 0
    w przeciwnym razie jeżeli n == 1 wykonaj:
        zwróć 1
    w przeciwnym razie:
        zwróć rekurencyjnieDrugie(n-1) * rekurencyjnieDrugie(n-2)
```

Funkcje napisane techniką rekurencyjną są czytelne i proste. Podobnie jak w przypadku problemu 1, jeżeli funkcja przyjmie parametr 0 lub 1, od razu zwracana jest konkretna wartość, natomiast dla innych parametrów, następują dwa wywołania rekurencyjne funkcji.

W przypadku **techniki iteracyjnej**, kod będzie dłuższy i trudniejszy do zrozumienia:

```
funkcja iteracyjnieDrugie(n)
    a = 0
    b = 1
    jeżeli n == 0 wykonaj:
        zwróć a
    dla i = 2, 3, ..., n wykonuj:
        c = a * b + 3
        a = b
        b = c
    zwróć b
```

Ponownie w przypadku zerowego i pierwszego wyrazu ciągu bezpośrednio zwracana jest wartość. Dla kolejnych wyrazów muszą zostać wykonane instrukcje w pętli. W zmiennej `c` umieszczany jest iloczyn dwóch poprzednich wyrazów ciągu, powiększony o 3, następnie w zmiennej `a` będzie przechowywana wartość poprzedniego wyrazu ciągu, która dotychczas była w zmiennej `b`, natomiast do zmiennej `b` zostaje zapisana wartość aktualnego wyrazu.

Rekurencja czy iteracja?

W przypadku problemu 1 algorytm możemy zapisać zarówno w wersji rekurencyjnej, jak i iteracyjnej. Jeżeli chodzi o problem 2, mimo iż wersja iteracyjna jest mniej czytelna, to właśnie ją lepiej jest zastosować. Technika rekurencyjna rozwiązywania zadania drugiego wymaga wielu wywołań rekurencyjnych, co jest bardzo nieefektywne obliczeniowo. Dla piątego wyrazu ciągu funkcja `rekurencyjnieDwa` wywoływana jest aż 15 razy!

Słownik

> **algorytm iteracyjny**
>
> ciąg operacji, który jest powtarzany tak długo, dopóki nie zostanie spełniony określony warunek
> **algorytm rekurencyjny**
>
> algorytm, w którym funkcja lub procedura wywołuje samą siebie aż do napotkania przypadku podstawowego
> **iteracja**
>
> słowo pochodzące od łacińskiego *iteratio* (powtarzanie); oznacza powtarzanie w pętli tych samych instrukcji, aż do spełnienia pewnego warunku
> **rekurencja**
>
> sytuacja, w której funkcja wywołuje samą siebie aż do napotkania przypadku podstawowego
> **silnia**
>
> $n!$ – iloczyn liczb naturalnych, mniejszych lub równych n

Symulacja interaktywna

Polecenie 1

Przeanalizuj symulację interaktywną, wizualizującą kolejne kroki algorytmu obliczania wartości elementów ciągu za pomocą dwóch metod: rekurencyjnej oraz iteracyjnej. Spróbuj, na kartce, samodzielnie wykonać przedstawione kroki dla ciągu:

$$a_n = \begin{cases} 1 & \text{gdy } n = 1 \\ 1 & \text{gdy } n = 2 \\ a_{n-1} + a_{n-2} & \text{gdy } n > 2 \end{cases}$$

Zasób interaktywny dostępny pod adresem https://zpe.gov.pl/a/D1970GVVp

Polecenie 2

Przygotuj notatkę ze swoimi spostrzeżeniami dotyczącymi obu metod przedstawionych w symulacji.

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Ćwiczenie 2

Napisz za pomocą pseudokodu funkcję realizującą algorytm obliczania silni liczby n metodą rekurencyjną.

Specyfikacja problemu:

Dane:

- n – liczba naturalna

Wynik:

Program wyświetla silnię liczby n.

Ćwiczenie 3

Wykorzystując pseudokod, napisz funkcję realizującą algorytm obliczania silni liczby n metodą iteracyjną.

Specyfikacja problemu:

Dane:

- n – liczba naturalna

Wynik:

Program wyświetla silnię liczby n.

Ćwiczenie 4

Za pomocą pseudokodu przedstawiona została funkcja realizująca algorytm rekurencyjnego wyznaczania wartości n-tego wyrazu ciągu. Rozwiąż przedstawiony problem, stosując technikę iteracyjną.

Specyfikacja problemu:

Dane:

- n – liczba naturalna; indeks elementu ciągu

Wynik:

Program wyświetla n-ty element danego ciągu.

```
Rekurencyjnie(n)
    jeżeli n = 1
        zwróć 1
    jeżeli n = 2
        zwróć 3
    jeżeli n = 3
        zwróć 2
    zwróć Rekurencyjnie(n - 1) + Rekurencyjnie(n - 3) - 1
```

Ćwiczenie 5

Ćwiczenie 6

Zapoznaj się z pseudokodem i wykonaj ćwiczenie.

```
Rekurencyjnie(n)
    jeżeli n = 1
        zwróć 1
    w przeciwnym wypadku
        zwróć Rekurencyjnie(n - 1) + 1
```

Ćwiczenie 7

Zapoznaj się z pseudokodem i wykonaj ćwiczenie.

```
Rekurencyjnie(n)
    jeżeli n = 1
        zwróć 1
    jeżeli n = 2
        zwróć 3
    jeżeli n = 3
        zwróć 2
    w przeciwnym wypadku
        zwróć Rekurencyjnie(n - 1) + Rekurencyjnie(n - 3) - 1
```

Ćwiczenie 8

Za pomocą pseudokodu zapisana została funkcja realizująca algorytm iteracyjnego wyznaczania wartości n-tego wyrazu ciągu. Rozwiąż przedstawiony problem, stosując technikę rekurencyjną.

Specyfikacja problemu:

Dane:

- n – liczba naturalna; indeks elementu ciągu

Wynik:

Program wyświetla n-ty element danego ciągu.

```
Iteracyjnie(n)
    a = 5
    dla i = 2, 3, …, n wykonuj
        a = -a - 4
    zwróć a
```

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Rekurencja a iteracja

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres podstawowy i rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

2) stosuje przy rozwiązywaniu problemów z różnych dziedzin algorytmy poznane w szkole podstawowej oraz algorytmy:

e) obliczania wartości elementów ciągu metodą iteracyjną i rekurencyjną, w tym wartości elementów ciągu Fibonacciego.

I + II. Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

3) objaśnia, a także porównuje podstawowe metody i techniki algorytmiczne oraz struktury danych, wykorzystując przy tym przykłady problemów i algorytmów, w szczególności:

b) rekurencję (do generowania ciągów liczb, potęgowania, sortowania liczb, generowania fraktali),

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Podsumujesz podstawowe informacje o iteracji i rekurencji.
- Wyjaśnisz, czy można zastąpić rekurencję iteracją i odwrotnie.
- Przeanalizujesz i porównasz algorytmy rekurencyjny i iteracyjny obliczania wartości elementów ciągu.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- metody aktywizujące;
- burza mózgów;
- mapa myśli.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Rekurencja a iteracja”. Uczniowie mają zapoznać się z treściami w sekcji „Przeczytaj”.

Faza wstępna:

1. Ustalenie celu lekcji i kryteriów sukcesu.
2. Prowadzący prosi uczniów, aby zgłaszali swoje propozycje pytań do tematu. Jedna osoba może zapisywać je na tablicy. Gdy uczniowie wyczerpią swoje pomysły, a pozostały jakieś ważne kwestie do poruszenia, nauczyciel je dopowiada.

Faza realizacyjna:

1. Uczniowie metodą burzy mózgów przypominają sobie najważniejsze informacje związane z iteracją oraz rekurencją. Chętne lub wybrane osoby podają przykłady problemów, które można rozwiązać tak iteracyjnie, jak rekurencyjne. Inne osoby podają z kolei przykłady problemów, które nie pozwalają na wymienne wykorzystywanie dwóch metod. Uczniowie zapisują efekty burzy mózgów za pomocą mapy myśli.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Symulacja interaktywna". Uczniowie wspólnie analizują symulację interaktywną wizualizującą kolejne kroki algorytmu obliczania wartości elementów ciągu za pomocą dwóch metod – rekurencyjnej i iteracyjnej. Następnie próbują samodzielnie wykonać poszczególne kroki dla ciągu przedstawionego w poleceniu nr 1.
3. **Ćwiczenie umiejętności.** Poszukiwanie najefektywniejszego rozwiązania problemu. Uczniowie wykonują w indywidualnie ćwiczenia nr 1–7 z sekcji „Sprawdź się", a następnie omawiają je na forum. Nauczyciel ocenia efektywność zastosowanego rozwiązania.

Faza podsumowująca:

1. Nauczyciel pyta uczniów o to, czy po zajęciach chcą uzupełnić mapę myśli.

Praca domowa:

1. Uczniowie wykonują ćwiczenie nr 8 z sekcji „Sprawdź się".

Wskazówki metodyczne:

- Treści w sekcji „Przeczytaj" można wykorzystać jako podsumowanie i utrwalenie wiedzy uczniów.