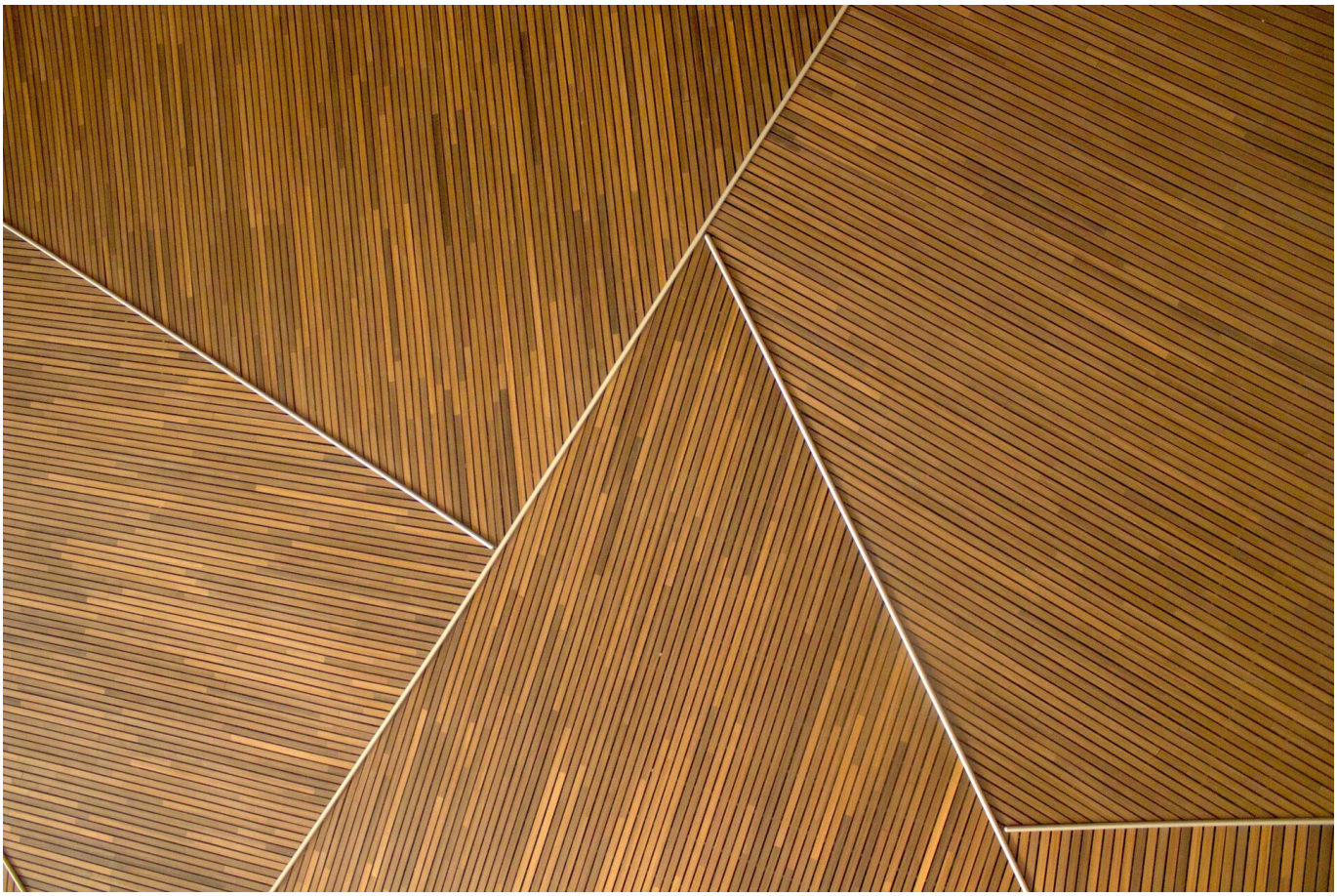

Algorytmy iteracyjne – zadania maturalne

Źródło: Teo Duldulao, domena publiczna.

Jak już wiemy z e-materiału Algorytmy iteracyjne, iteracja to powtarzanie zapisanych w programie instrukcji w pętli z góry określoną ilość razy lub aż do spełnienia określonego warunku. Mechanizm ten jest podstawową operacją wykorzystywaną w programowaniu.

W tym e-materiale zapoznamy się z przykładowymi zadaniami maturalnymi dotyczącymi tego zagadnienia.

Implementacje algorytmów iteracyjnych w poszczególnych językach programowania przedstawiamy w e-materiałach:

- Algorytmy iteracyjne w języku C++,
- Algorytmy iteracyjne w języku Java,
- Algorytmy iteracyjne w języku Python.

Twoje cele

- Przeanalizujesz działanie algorytmów iteracyjnych.
- Rozwiążesz samodzielnie zadanie maturalne, wykorzystujące algorytmy iteracyjne.
- Przeanalizujesz zasady oceniania zadań maturalnych.

Przeczytaj

Zadania maturalne

Zadanie 1.

Rozważmy następujący algorytm:

Specyfikacja problemu:

Dane:

- `n` – liczba całkowita dodatnia

Wynik:

- `p` – liczba całkowita dodatnia

```
p <- 1
q <- n
dopóki p < q wykonuj
    s <- (p + q) div 2
    jeżeli s * s * s < n wykonuj
        p <- s + 1
    w przeciwnym wypadku
        q <- s
```

> **Ważne!**
>
> Zapis `div` oznacza dzielenie całkowite.

> Zadanie zostało opracowane przez Centralną Komisję Egzaminacyjną i pojawiło się na egzaminie maturalnym z informatyki w maju 2018 r. (poziom rozszerzony,

> część I, formuła od 2015 r.). Cały arkusz można znaleźć na stronie internetowej CKE.

Zadanie 1.1. (0-3 pkt)

Treść polecenia

Podaj wynik działania algorytmu dla wskazanych w tabeli wartości `n`.

n	p
28	?
64	?
80	?

Rozwiązanie

W tym zadaniu należy przeanalizować algorytm zapisany za pomocą pseudokodu, a następnie podać wyniki w zależności od danych. Prześledźmy algorytm krok po kroku dla każdej z podanych wartości `n`.

Przygotowujemy tabelę, w której będziemy zapisywać kolejne wartości zmiennych `p`, `q`, `s`.

```
n = 28
```

p	q	s
1	28	?

```
1 < 28  PRAWDA
    s = (1+28) div 2 = 14
    14*14*14 < 28   FAŁSZ
        q = 14
```

---

p	q	s
1	14	14

```
1 < 14  PRAWDA
    s = (1+14) div 2 = 7
    7*7*7 < 28  FAŁSZ
        q = 7
```

p	q	s
1	7	7

```
1 < 7   PRAWDA
    s = (1+7) div 2 = 4
    4*4*4 < 28  FAŁSZ
        q = 4
```

p	q	s
1	4	4

```
1 < 4   PRAWDA
    s = (1+4) div 2 = 2
    2*2*2 < 28  PRAWDA
        p = 3
```

p	q	s
3	4	2

```
3 < 4   PRAWDA
    s = (3+4) div 2 = 3
    3*3*3 < 28  PRAWDA
        p = 4
```

```
5
6 4 < 4   FAŁSZ
```

p	q	s
4	4	3

n = 64

p	q	s
1	64	?

```
1 p = 1
2 q = 64
3
4 1 < 64  PRAWDA
5     s = (1+64) div 2 = 32
6     32*32*32 < 64   FAŁSZ
7         q = 32
```

p	q	s
1	32	32

```
1 1 < 32  PRAWDA
2     s = (1+32) div 2 = 16
3     16*16*16 < 64   FAŁSZ
4         q = 16
```

p	q	s
1	16	16

```
1 < 16  PRAWDA
    s = (1+16) div 2 = 8
    8*8*8 < 64      FAŁSZ
        q = 8
```

p	q	s
1	8	8

```
1 < 8   PRAWDA
    s = (1+9) div 2 = 5
    5*5*5 < 64      FAŁSZ
        q = 5
```

p	q	s
1	5	5

```
1 < 5   PRAWDA
    s = (1+5) div 2 = 3
    3*3*3 < 64      PRAWDA
        p = 3+1 = 4
```

p	q	s
4	5	3

```
4 < 5   PRAWDA
    s = (4+5) div 2 = 4
    4*4*4 < 64      FAŁSZ
        q = 4

4 < 4   FAŁSZ
```

p	q	s
4	4	4

n = 80

p	q	s
1	80	?

```
p = 1
q = 80

1 < 80  PRAWDA
    s =  (1+80) div 2 = 40
    40*40*40 < 80   FAŁSZ
        q = 40
```

p	q	s
1	40	40

```
1 < 40  PRAWDA
    s = (1+40) div 2 = 20
    20*20*20 < 80   FAŁSZ
        q = 20
```

p	q	s
1	20	20

```
1 < 20  PRAWDA
```

```
    s = (1+20) div 2 = 10
    10*10*10 < 80   FAŁSZ
        q = 10
```

p	q	s
1	10	10

```
1 < 10  PRAWDA
    s = (1+10) div 2 = 5
    5*5*5 < 80      FAŁSZ
        q = 5
```

p	q	s
1	5	5

```
1 < 5   PRAWDA
    s = (1+5) div 2 = 3
    3*3*3 < 80      PRAWDA
        p = 4
```

p	q	s
4	5	3

```
4 < 5   PRAWDA
    s = (4+5) div 2 = 4
    4*4*4 < 80      PRAWDA
        p = 5

5 < 5   FAŁSZ

```

p	q	s
5	5	4

Schemat oceniania

Rozwiązywanie problemów i podejmowanie decyzji [...], stosowanie podejścia algorytmicznego. Zdający:
11) opisuje podstawowe algorytmy i stosuje:
a) algorytmy na liczbach całkowitych;
16) opisuje własności algorytmów na podstawie ich analizy;
17) ocenia zgodność algorytmu ze specyfikacją problemu;
18) oblicza liczbę operacji wykonywanych przez algorytm.

Poprawna odpowiedź

n	p
28	4
64	4
80	5

3 pkt – za prawidłową odpowiedź w trzech wierszach.

2 pkt – za prawidłową odpowiedź w dwóch wierszach.

1 pkt – za prawidłową odpowiedź w jednym wierszu.

0 pkt – za podanie odpowiedzi błędnej albo brak odpowiedzi.

Zadanie 1.2. (0-2 pkt)

Podaj najmniejszą oraz największą liczbę `n`, dla której wynikiem działania algorytmu będzie `p = 10`.

Rozwiązanie

W tym zadaniu należy dokładnie przeanalizować wyniki, jakie zwraca program w poprzednim podpunkcie. Zwróć uwagę, że zawsze sprawdzamy, czy dana wartość

n jest większa od s^3.

Spójrzmy teraz na wyniki programu dla kolejnych wartości n.

n	p
28	4
64	4
80	5

Jeżeli wartość zmiennej s podniesionej do trzeciej potęgi była mniejsza od n, to zmiennej p przypisywaliśmy wartość inkrementowanej zmiennej s.

Sprawdźmy teraz, ile wynoszą sześciany dla wartości p, które były odpowiedzią w poprzednim zadaniu i porównajmy je z wartościami sześcianu dla s mniejszego o 1 od p.

n	p^3	s^3
28	64	27
64	64	27
80	125	64

Jak możemy zaobserwować, między zmiennymi zachodzi następująca zależność:

$$s^3 < n <= p^3.$$

Wynika stąd, że aby otrzymać wynik $p = 10$, wartość zmiennej n musiałaby należeć do przedziału:

$$729 < n \leq 1000.$$

Pamiętajmy też, że:

$$s = p - 1.$$

W konsekwencji, ponieważ n jest liczbą naturalną, to najmniejsza liczba wynosi 730, a największa 1000.

Schemat oceniania

Rozwiązywanie problemów i podejmowanie decyzji [...], stosowanie podejścia algorytmicznego. Zdający:
11) opisuje podstawowe algorytmy i stosuje:
a) algorytmy na liczbach całkowitych;
16) opisuje własności algorytmów na podstawie ich analizy;
17) ocenia zgodność algorytmu ze specyfikacją problemu;
18) oblicza liczbę operacji wykonywanych przez algorytm.

Poprawna odpowiedź

`730`, `1000`.

2 pkt – za prawidłową odpowiedź w dwóch wierszach.

1 pkt – za prawidłową odpowiedź w jednym wierszu.

0 pkt – za podanie odpowiedzi błędnej albo brak odpowiedzi.

Słownik

> **inkrementacja**
>
> zwiększenie wartości zmiennej o 1

Symulacja interaktywna

Polecenie 1

Zapoznaj się z algorytmem zapisanym za pomocą pseudokodu. Tablica T przechowuje dowolne liczby całkowite.

a. Uzupełnij specyfikację, zapisując wynik działania podanego kodu.

Specyfikacja problemu:

Dane:

- n – liczba elementów w tablicy T; liczba naturalna
- `T[0, ..., n-1]` – tablica liczb całkowitych

Wynik:

...

Pokaż rozwiązanie

Posortowana niemalejąco tablica T

```
dla i = 1, 2, ..., n - 1 wykonuj:
  p = T[i]
  j = i - 1
  dopóki j >= 0 oraz T[j] > p wykonuj:
      T[j + 1] = T[j]
      j = j - 1
  T[j + 1] = p

```

b. Dla tablicy `T = {9, 3, 6, 5, 7, 2, 6}` uzupełnij zawartość tablicy T po wykonaniu kolejnych iteracji pętli zewnętrznej.

i	**0**	**1**	**2**	**3**	**4**	**5**	**6**
–	9	3	6	5	7	2	6
1	3	9	6	5	7	2	6
2							

i	0	1	2	3	4	5	6
3							
4							
5							
6							

Polecenie 2

Symulacja interaktywna ukazuje działanie algorytmu z polecenia 1 w formie schematu blokowego.

Zasób interaktywny dostępny pod adresem https://zpe.gov.pl/a/D5gTaAUW2

Polecenie 3

Zaimplementuj w wybranym języku programowania algorytm z polecenia 2.

```
1
```

Sprawdź się

Pokaż ćwiczenia:

Materiał źródłowy do ćwiczeń nr 1–2.

Przeanalizuj poniższy algorytm i wykonaj ćwiczenia.

```
jeżeli n < 3
    wynik = 1
w przeciwnym wypadku
    a = 1
    wynik = 1
  * dopóki n - 2 >= 0 wykonuj
        pomoc = wynik
        wynik = wynik + a
        a = pomoc
        n = n - 1

```

Ćwiczenie 1

Wskaż, ile razy wykona się pętla oznaczona gwiazdką dla określonych wartości n.

n=2	0
n=100	99
n=4	19
n=20	8
n=9	3

Ćwiczenie 2

Uzupełnij tabelę, zapisując wyniki działania algorytmu dla danych wartości n

n	wynik
2	
5	
7	
12	

Materiał źródłowy do ćwiczeń nr 3–5.

> Algorytm zapisany za pomocą pseudokodu pochodzi z zadania o numerze 8 (wiązka zadań *Dwa ciągi)* znajdującego się w zbiorze zadań z informatyki autorstwa Centralnej Komisji Egzaminacyjnej. Cały zbiór można pobrać ze strony internetowej CKE.
>
> Pozostałe zadania zostały ułożone na podstawie wyżej wspomnianego zadania.

Specyfikacja problemu:

Dane:

- `n` – liczba naturalna dodatnia; rozmiar tablicy
- `A[1, n], B[1, n]` – uporządkowane rosnąco tablice liczb całkowitych
- `x` – liczba naturalna

Wynik:

Program drukuje wynik PRAWDA, gdy istnieje suma złożona z dowolnej liczby z tablicy `A` i dowolnej liczby z tablicy `B`, która jest równa liczbie `x`.

Program drukuje wynik FAŁSZ, gdy nie ma takiej pary.

```
i=1
j=n
dopóki i <= n oraz j >= 1 wykonuj
(*) jeżeli A[i] + B[j] == x wykonuj
            podaj wynik PRAWDA i zakończ algorytm
    w przeciwnym razie
        jeżeli A[i] + B[j] < x wykonuj
            i = i + 1
        w przeciwnym razie
            j = j – 1
podaj wynik FAŁSZ
```

Ćwiczenie 3

Uzupełnij tabelę.

nr wiersza	tablica A	tablica B	x	n	wynik działania algorytmu
1	3, 8, 9, 12	16, 20, 21, 30	30	4	
2	4, 6, 10	1, 7, 9	12	3	
3	11, 25, 32, 46, 49	6, 9, 10, 13, 17	55	5	
4	23, 34, 41, 45, 56, 60, 62	13, 20, 23, 32, 36, 39, 43	92	7	

Ćwiczenie 4

Uzupełnij tabelę. Ustal, ile porównań oznaczonych (*) zostanie wykonanych dla danych z tabeli z ćwiczenia 3.

nr wiersza	liczba porównań (*)
1	
2	
3	
4	

Ćwiczenie 5

Uzupełnij specyfikację algorytmu za pomocą podanych wyrażeń.

Specyfikacja problemu:

Dane:

n – [ ]

A[ ], B[ ] – tablice zawierające [ ] liczb całkowitych [ ]

x – [ ]

Wynik:

PRAWDA, gdy [ ]

FAŁSZ, gdy [ ]

posortowanych rosnąco | x – 1 | nie ma takiej liczby x w żadnej z tablic | suma dwóch elementów z tablicy A[ ] i B[ ] równa jest x | x | suma dwóch elementów z tablicy A[ ] i B[ ] równa jest n | nie ma takiej sumy dwóch elementów z tablicy A[ ] i B[ ], która równa jest n | nie ma takiej liczby n w żadnej z tablic | jest taka liczba n w jednej z tablic | jest taka liczba x w jednej z tablic | posortowanych malejąco | nie ma takiej sumy dwóch elementów z tablicy A[ ] i B[ ], która równa jest x | n − 1 | liczba całkowita | n | rozmiar tablic

Materiał źródłowy do ćwiczeń nr 6–7.

```
a1 = 2
a2 = 4
i = 3
jeżeli n >= 2
    wypisz(a1)
    wypisz(a2)
```

```
    dopóki i <= n wykonuj
        pomoc = a2
        a2 = a1 + 2 * a2
        a1 = pomoc
        wypisz(a2)
        i = i + 1
```

Ćwiczenie 6

Wskaż, co wypisze program dla n = 7.

- ◯ 2, 4, 8, 16, 32, 64, 128
- ◯ 2, 4, 6, 10, 16, 26, 422
- ◯ 2, 4, 12, 32, 88, 240
- ◯ 2, 4, 10, 24, 58, 140, 338

Ćwiczenie 7

Wskaż, jaki jest wzór na n-ty wyraz ciągu liczbowego, generowanego przez podany algorytm zapisany za pomocą pseudokodu.

- ◯ $a_n = 2a_{n-2} - a_{n-1}$
- ◯ $a_n = a_{n-2} + 2a_{n-1}$
- ◯ $a_n = 2a_{n-2} + a_{n-1}$
- ◯ $a_n = a_{n-2} + a_{n-1}$

Materiał źródłowy do ćwiczeń nr 8–9.

Rozważmy następującą procedurę, której parametrem jest dodatnia liczba całkowita n.

Zadania zostały opracowane przez CKE i pojawiły się na egzaminie maturalnym z informatyki w czerwcu 2017 roku (poziom rozszerzony, część I). Cały arkusz można znaleźć na stronie internetowej Centralnej Komisji Egzaminacyjnej.

```
1  Procedura Sitko(n)
2  dla i = 1, 2, ..., n wykonuj
3      Czyjest[i] = fałsz
4  j = 1
5  dopóki j * j < n wykonuj
6      j = j + 1
7  dla i = 2, 3, ..., j wykonuj
8      kw = i * i
9      poz = kw
10     dopóki poz ≤ n wykonuj
11 (*)     Czyjest[poz] = prawda
12         poz = poz + kw
```

Ćwiczenie 8

Uzupełnij tabelę. Wpisz wartości zmiennych j oraz `Czyjest[k]` po wykonaniu `Sitko(n)`.

n	k	j	Czyjest[k]
10	9	4	prawda
10	5		
100	10		
100	75		

Ćwiczenie 9

Podaj liczbę wykonań instrukcji w wierszu oznaczonym (*) – dla wartości zmiennej i wskazanych w tabeli z ćwiczenia 8. Rozważmy działanie `Sitko(100)`.

`i`	liczba wykonań wiersza (*)
2	25
3	
5	
9	

Dla nauczyciela

Autor: Maurycy Gast

Przedmiot: Informatyka

Temat: Algorytmy iteracyjne – zadania maturalne

Grupa docelowa:

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

Podstawa programowa:

Cele kształcenia – wymagania ogólne

> 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

> II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.
>
> Zakres podstawowy. Uczeń:
>
> 1) projektuje i programuje rozwiązania problemów z różnych dziedzin, stosuje przy tym: instrukcje wejścia/wyjścia, wyrażenia arytmetyczne i logiczne, instrukcje warunkowe, instrukcje iteracyjne, funkcje z parametrami i bez parametrów, testuje poprawność programów dla różnych danych; w szczególności programuje algorytmy z punktu I.2);

2) do realizacji rozwiązań problemów prawidłowo dobiera środowiska informatyczne, aplikacje oraz zasoby, wykorzystuje również elementy robotyki;

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

1) projektuje i tworzy rozbudowane programy w procesie rozwiązywania problemów, wykorzystuje w programach dobrane do algorytmów struktury danych, w tym struktury dynamiczne i korzysta z dostępnych bibliotek dla tych struktur;

2) stosuje zasady programowania strukturalnego i obiektowego w rozwiązywaniu problemów;

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

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):

- Przeanalizujesz działanie algorytmów iteracyjnych.
- Rozwiążesz samodzielnie zadanie maturalne, wykorzystujące algorytmy iteracyjne.
- Przeanalizujesz zasady oceniania zadań maturalnych.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- 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;
- oprogramowanie dla języka C++, w tym kompilator GCC/G++ 4.5 (lub nowszej wersji) i Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio;
- oprogramowanie dla języka Java SE 8 (lub nowszej wersji), w tym Eclipse 4.4 (lub nowszej wersji);
- oprogramowanie dla języka Python 3 (lub nowszej wersji), w tym PyCharm lub IDLE.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Algorytmy iteracyjne – zadania maturalne". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla uczniom temat zajęć oraz cele. Prosi, by na ich podstawie uczniowie sformułowali kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel zadaje uczniom pytanie dotyczące ich aktualnego stanu wiedzy w obszarze poruszanego tematu, opartego o programowanie.

Faza realizacyjna:

1. Uczniowie pracując w parach, analizują przykład z sekcji „Przeczytaj" i implementują przedstawiony algorytm w wybranych przez siebie językach programowania.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Symulacja interaktywna", wybrany uczeń czyta treść polecenia 1 i omawia przykładowe rozwiązanie postawionego problemu. Następnie uczniowie zapoznają się z symulacją interaktywną.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują ćwiczenia nr 1-7 z sekcji „Sprawdź się". Nauczyciel sprawdza poprawność wykonanych zadań, omawiając je wraz z uczniami.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności, omawia ewentualne problemy podczas rozwiązywania ćwiczeń.

Praca domowa:

1. Uczniowie wykonują ćwiczenie 8-9 z sekcji „Sprawdź się".
2. Uczniowie implementują algorytm z sekcji „Symulacja interaktywna" w wybranych przez siebie językach programowania.

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka C++.
- Oficjalna dokumentacja techniczna dla kompilatora GCC/G++ 4.5 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.
- Oficjalna dokumentacja techniczna dla języka Java SE 8 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Eclipse 4.4 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla języka Python 3 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania PyCharm lub IDLE.

Wskazówki metodyczne:

- Treści w sekcji „Symulacja interaktywna” można wykorzystać na lekcji jako podsumowanie i utrwalenie wiedzy uczniów.