Algorytmika – zadania maturalne

Źródło: Rodion Kutsaev, dostępny w internecie: unsplash.com, domena publiczna.

Egzamin maturalny z informatyki ma dokładnie sprecyzowaną strukturę i formę zadań. W tym e-materiale zobaczysz, jak wygląda przykładowe zadanie dotyczące zagadnień związanych z algorytmiką. Przedstawimy również, z jakiego typu zadaniami możesz się spotkać, i jak będą punktowane.

Informacje o algorytmach znajdziesz m.in. w pozostałych e-materiałach z serii:

- Algorytmika,
- Algorytmy na co dzień,
- Własności algorytmów,
- Specyfikacja problemu algorytmicznego.

Twoje cele

- Przeanalizujesz strukturę egzaminu maturalnego z informatyki.
- Omówisz typy zadań teoretycznych występujących na egzaminie maturalnym z informatyki.
- Rozwiążesz zadania z arkuszy maturalnych dotyczące algorytmów.

Przeczytaj

> **Ważne!**
>
> Informacje dotyczące egzaminu maturalnego z informatyki pochodzą z *Informatora o egzaminie maturalnym z informatyki od roku szkolnego 2022/2023* dostępnego na stronie internetowej CKE.

Egzamin maturalny z informatyki

W arkuszu egzaminacyjnym znajdą się zarówno zadania zamknięte, jak i otwarte oraz praktyczne.

W arkuszu egzaminacyjnym część zadań wymaga jedynie zapisu odpowiedzi w arkuszu papierowym. Rozwiązanie większości zadań wymaga użycia komputera i zapisania komputerowej realizacji rozwiązania. W zadaniach tego typu rozwiązanie, w którym brakuje komputerowej realizacji, nie będzie oceniane.

Typy zadań i zasady oceniania

Zadania zamknięte i zadania otwarte z luką

Zadania zamknięte są oceniane – w zależności od maksymalnej liczby punktów, jaką można uzyskać za rozwiązanie danego zadania – zgodnie z następującymi zasadami:

- 1 pkt – odpowiedź poprawna.
- 0 pkt – odpowiedź niepełna albo odpowiedź niepoprawna albo brak odpowiedzi.

Zadania z otwartą luką są oceniane zgodnie z następującymi zasadami:

- 2 pkt – odpowiedź całkowicie poprawna.
- 1 pkt – odpowiedź częściowo poprawna lub odpowiedź niepełna.
- 0 pkt – odpowiedź niepoprawna albo brak odpowiedzi.

Przykładowe zadania

> Na podstawie specyfikacji algorytmu oraz algorytmu wykonaj polecenia.
>
> Algorytm obliczania n-tego wyrazu ciągu geometrycznego.
>
> **Specyfikacja problemu:**
>
> *Dane:*

- `a` – liczba rzeczywista oznaczająca pierwszy wyraz ciągu
- `q` – liczba rzeczywista będąca ilorazem ciągu
- `n` – liczba naturalna oznaczająca numer wyrazu ciągu, którego wartość chcemy policzyć

Wynik:

- `w` – liczba rzeczywista, będąca `n`-tym wyrazem ciągu

Polecenie 1

Podaj wynik działania algorytmu, jeśli `a` jest równe 5, `q` jest równe 5 i `n` jest równe 2.

```
w = a
dla i = 0, 1, 2, ..., n-1 wykonuj:
    w = w * q
zwróć w
```

Należy prześledzić działanie algorytmu. Dokonujemy obliczeń w miejscu wyznaczonym w treści zadania. W tym przypadku odpowiedzią jest 125.

W zadaniach tego typu, zazwyczaj, jest jeszcze tabela do uzupełnienia.

Polecenie 2

Dla danych a, q, n i powyższego algorytmu uzupełnij tabelę.

```
w = a
dla i = 0, 1, 2, ..., n-1 wykonuj:
    w = w * q
zwróć w
```

a	q	n	w
0	5	8	$\ldots$
5	$\ldots$	2	125
2	4	3	$\ldots$
$\ldots$	4	4	256
4	2	5	$\ldots$
5	0	9	$\ldots$

Treść zadania, co do zasady, jednoznacznie wskazuje, że każdy wiersz tabeli stanowi oddzielne uruchomienie algorytmu dla podanych danych. Każde oddzielne uruchomienie

> musimy w pewien sposób rozpisać i zapisać wynik w odpowiedniej komórce. Kolumny, które należy uzupełnić, zależą od poziomu trudności zadania. Wymagane może być wypełnienie jedynie kolumny `w`. Możliwa jest też inna konfiguracja, w której mamy `w` oraz `q`, a nie znane jest `a`. Bądź też taka, w której mamy: `a` i `w`, lecz nie mamy `q`. Spróbuj rozwiązać to zadanie samodzielnie.

Kluczowymi kwestiami podczas rozwiązywania tego typu zadań są zrozumienie algorytmu oraz umiejętność jego uogólnienia. Jest to bardzo ważne, ponieważ od zrozumienia działania algorytmu zależy poprawność rozwiązania kolejnych poleceń do danego zadania.

Polecenie 3

Zaznacz zdania prawdziwe o dotyczące sortowań.

Zdanie	P/F
Algorytm sortowania bąbelkowego jest algorytmem typu „dziel i zwyciężaj”.	P/F
Algorytm sortowania przez zliczanie ma niską złożoność pamięciową dla dużej liczby unikalnych wartości.	P/F
Algorytm sortowania kubełkowego to inaczej algorytm sortowania przez zliczanie.	P/F
Złożoność obliczeniowa $O\left(n^2\right)$, w kontekście sortowań, oznacza, że wraz z liczbą elementów do posortowania, ta rośnie wykładniczo.	P/F

Polecenie 4

Zaznacz zdania prawdziwe dotyczące algorytmów.

Zdanie	P/F
Złożoność układania wieży Hanoi rośnie wykładniczo wraz z ilością krążków.	P/F
W pesymistycznym przypadku w drzewie binarnym czas wyszukiwania wartości w danym drzewie wynosi $O(n)$.	P/F
Problem komiwojażera to problem, który polega na znalezieniu najkrótszej możliwej trasy pomiędzy wszystkimi wierzchołkami grafu.	P/F
Kolejka FIFO charakteryzuje się tym, że najpierw z kolejki wychodzą te obiekty, które jako ostatnie do niej weszły.	P/F

Polecenie 5

> Zadanie to pochodzi z arkusza I matury rozszerzonej z 2016 roku i dostępne jest na stronie CKE.

Dana jest funkcja f określona wzorem rekurencyjnym

$$\begin{cases} f(1)=4 \\ f(n\ +\ 1)=\frac{1}{1-f(n)}\ dla\ n\ \geq\ 1 \end{cases}$$

Wtedy:

Zdanie	**P/F**
$f(8) = \frac{1}{3}$	P/F
$f(9) = \frac{3}{4}$	P/F
$f(10) = 4$	P/F
$f(100) = -\frac{1}{3}$	P/F

Polecenie 6

> Zadanie to pochodzi z matury rozszerzonej z 2021 roku i dostępne jest na stronie CKE.

```
funkcja f(n):
    jeżeli n > 0
        wypisz n
        f(n - 2)
        wypisz n
```

Zdanie	P/F
W wyniku wywołania $f\,(5)$ otrzymamy ciąg $5\ 5\ 5\ 5\ 5\ 5$.	P/F
W wyniku wywołania $f\,(6)$ otrzymamy ciąg $6\ 4\ 2\ 2\ 4\ 6$.	P/F
W wyniku wywołania $f\,(7)$ otrzymamy ciąg $7\ 5\ 3\ 1\ 1\ 3\ 5\ 7$.	P/F
W wyniku wywołania $f\,(8)$ otrzymamy ciąg $8\ 6\ 4\ 2\ 0\ 0\ 2\ 4\ 6\ 8$.	P/F

Zadania otwarte krótkiej odpowiedzi

Za rozwiązanie zadania otwartego krótkiej odpowiedzi będzie można otrzymać od 0 do 3 punktów. Zasady oceniania będą opracowywane odrębnie dla każdego zadania.

Za każde poprawne rozwiązanie, inne niż opisane w zasadach oceniania, można przyznać maksymalną liczbę punktów, o ile rozwiązanie jest merytorycznie poprawne, zgodne z poleceniem i warunkami zadania.

Zadania otwarte rozszerzonej odpowiedzi

Za rozwiązanie zadania otwartego rozszerzonej odpowiedzi będzie można otrzymać od 0 do 5 punktów. Schemat oceniania będzie opracowywany odrębnie dla każdego zadania.

Na podstawie poniższej specyfikacji algorytmu wykonaj polecenia.

Algorytm obliczania n-tego wyrazu ciągu geometrycznego.

Specyfikacja problemu:

Dane:

- a – liczba całkowita dodatnia; pierwszy wyraz ciągu
- r – liczba całkowita dodatnia; różnica ciągu arytmetycznego
- n – liczba naturalna; liczba wyrazów ciągu, których sumę liczymy

Wynik:

- s – liczba całkowita dodatnia; suma n wyrazów ciągu arytmetycznego

Polecenie 7

Uzupełnij poniższy pseudokod, tak aby był zgodny ze specyfikacją algorytmu.

```
s = a

dla i = 2, … n wykonuj:

zwróć s
```

W algorytmie trzeba uzupełnić trzecią linijkę odpowiednim fragmentem kodu, lub wybrać gotowy fragment kodu ze wskazanej w samym zadaniu puli możliwości. W tym przypadku poprawne rozwiązanie wygląda następująco:

```
s = s + a + r * (i - 1)
```

Odpowiedź:

```
s = a

dla i = 2, … n wykonuj:
    s = s + a + r * (i - 1)
zwróć s
```

Polecenie 8

Napisz za pomocą pseudokodu lub wybranego przez siebie języka programowania algorytm, który obliczy silnię liczby naturalnej n.

Specyfikacja problemu:

Dane:

- n – liczba naturalna

Wynik:

- `silnia` – silnia liczby naturalnej n

> W takim zadaniu najczęściej znajduje się zilustrowane przykładami wyjaśnienie, czym jest silnia.

Silnia liczby naturalnej n jest iloczynem wszystkich liczb z przedziału od 1 do n i oznaczamy ją za pomocą znaku `!`, np.:

$$3! = 3 \cdot 2 \cdot 1,$$

$$25! = 25 \cdot 24 \cdot 23 \cdot 22 \cdot 21 \cdot \ldots 3 \cdot 2 \cdot 1,$$

Poprawne rozwiązanie tego zadania zapisane za pomocą pseudokodu mogłoby wyglądać następująco:

```
1 silnia = 1
2 dla i = 1, 2, 3, ..., n wykonuj:
3     silnia = silnia * i
4 zwróć silnia
```

W tym typie zadań, podobnie jak w typie pierwszym, bardzo istotna jest znajomość samych algorytmów, na podstawie których będzie się opracowywać rozwiązania.
W przypadku ostatniego zadania skorzystaliśmy z wiedzy, że silnia jest iloczynem wszystkich liczb naturalnych dodatnich nie większych od n, a z racji tego, że mnożenie jest przemienne, to jako ostateczny wynik mogliśmy zwrócić kwadrat z obliczonej silni.

Zadania praktyczne wymagające użycia komputera i zapisania komputerowej realizacji

Za rozwiązanie zadania praktycznego można będzie otrzymać od 0 do 4 punktów. Do oceny w takim zadaniu należy przekazać pliki zawierające komputerowe realizacje rozwiązań oraz odpowiedzi zapisane w pliku tekstowym lub w arkuszu egzaminacyjnym – zgodnie z treścią zadania.

W zadaniach tego typu oceniane są rzeczywiste efekty i osiągnięte rezultaty przez zdającego, tj.: wyniki obliczeń w arkuszu kalkulacyjnym, wyniki symulacji, odpowiedzi uzyskane za pomocą kwerend, wyniki działania programu napisanego przez zdającego. Dołączenie komputerowej realizacji zadania (czyli programu, arkusza kalkulacyjnego lub bazy danych) jest wymagane.

Słownik

> **algorytm**
>
> przepis postępowania prowadzący do rozwiązania ustalonego problemu, określający ciąg czynności elementarnych, które należy w tym celu wykonać
> **specyfikacja algorytmu**
>
> specyfikacja problemu algorytmicznego; opis problemu, który ma zostać rozwiązany wraz z danymi oraz wynikiem

Prezentacja multimedialna

Zadanie teoretyczne typu „ułóż algorytm”

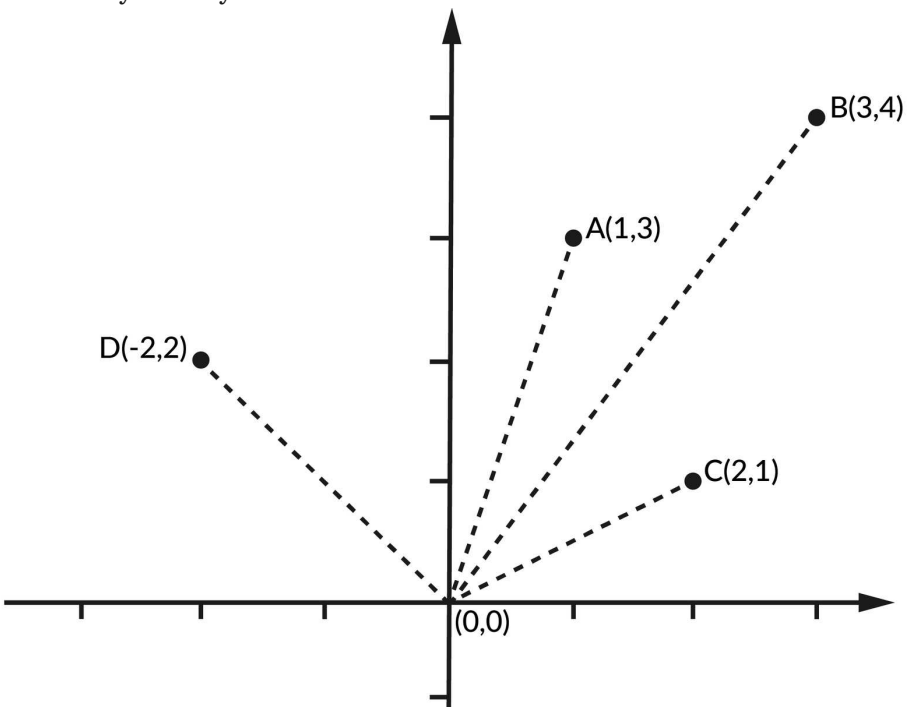
> Poniższe zadanie pochodzi z części I arkusza rozszerzonego egzaminu maturalnego z informatyki z maja 2018 roku. Jest dostępne w oryginalnej formie na oficjalnej stronie internetowej Centralnej Komisji Egzaminacyjnej.

W pewnym paśmie górskim znajduje się n szczytów, które będziemy przedstawiać jako punkty w układzie kartezjańskim na płaszczyźnie. Wszystkie punkty leżą powyżej osi OX, tzn. druga współrzędna (y) każdego punktu jest dodatnia.

W punkcie $(0,\ 0)$ stoi obserwator. Jeśli dwa szczyty A i B mają współrzędne $(x_A,\ y_A)$ oraz $(x_B,\ y_B)$, to mówimy, że:

- szczyt A jest dla obserwatora widoczny na lewo od B, jeśli $\frac{x_A}{y_A} < \frac{x_B}{y_B}$,
- szczyt B jest widoczny na lewo od A, jeśli $\frac{x_A}{y_A} > \frac{x_B}{y_B}$.

Wiemy, że żadne dwa szczyty nie leżą w jednej linii z obserwatorem, a zatem dla obserwatora te szczyty nie zasłaniają się nawzajem. Ilustrację przykładowego położenia szczytów można zobaczyć na rysunku:



Źródło: Contentplus.pl sp. z o.o., licencja: CC BY-SA 3.0.

W tym przykładzie, patrząc od lewej do prawej strony, obserwator widzi kolejno szczyt D, szczyt A, szczyt B i szczyt C.

Współrzędne szczytów dane są w dwóch tablicach `X[1..n]` oraz `Y[1..n]` – szczyt numer i ma współrzędne `(X[i], Y[i])`.

Polecenie 1

Napisz algorytm (w pseudokodzie lub wybranym języku programowania), który znajdzie i poda współrzędne *skrajnie lewego szczytu*, tzn. widocznego dla obserwatora na lewo od wszystkich pozostałych szczytów.

Specyfikacja problemu:

Dane:

- n – liczba całkowita dodatnia
- `X[0..n - 1]` – tablica liczb całkowitych
- `Y[0..n - 1]` – tablica liczb całkowitych dodatnich
- Para `(X[i], Y[i])` to współrzędne jednego szczytu, `i = 0, 1, ..., n - 1.`
- Żadne dwa szczyty nie leżą w jednej linii z obserwatorem.

Wynik:

- x, y – współrzędne skrajnie lewego szczytu spośród tych opisanych w tablicach X i Y

```
1
```

Polecenie 2

Porównaj swoje rozwiązanie z rozwiązaniem przedstawionym w prezentacji.

1

Na egzaminie maturalnym możesz spotkać się z wieloma typami zadań, np. zadaniami z luką, zadaniami krótkiej odpowiedzi oraz zadaniami rozszerzonej odpowiedzi. Są to zadania otwarte. Na egzaminie pojawiają się jednak również zadania zamknięte. Prezentowane zadanie jest zadaniem otwartym.
Źródło: Pixabay, CC0.

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Zastanówmy się, jaką postać przyjmie wynik.

Zgodnie z poleceniem są to dwie wartości: x i y, które reprezentują współrzędne najbardziej skrajnie lewego szczytu. Mamy podać tylko jeden wierzchołek. Treść zadania określa sposób porównywania, który z wierzchołków jest tym właściwym: „szczyt A jest dla obserwatora widoczny na lewo od B, jeśli $\frac{x_A}{y_A} < \frac{x_B}{y_B}$".

Podano tę informację drugi raz, tym razem ze sprawdzaniem, czy punkt A jest bardziej w skrajnym lewym rogu niż punkt B:

„szczyt B jest widoczny na lewo od A, jeśli $\frac{x_A}{y_A} > \frac{x_B}{y_B}$".

Oznacza to, że bardziej skrajnym punktem po lewej jest ten, który dla $\frac{x_N}{y_N}$ daje nam mniejszy wynik.

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Spróbujmy wyjaśnić, dlaczego tak jest.

Poruszamy się po pierwszej i drugiej ćwiartce układu współrzędnych. Oznacza to, że współrzędna y nigdy nie będzie ujemna. Wynik dzielenia współrzędnej x przez y może jednak być liczbą ujemną, w zależności od wartości zmiennej x.

Ważne jest, że nie trzeba będzie dzielić dwóch liczb ujemnych. Dzięki temu nie uwzględniamy sytuacji, w której najbardziej lewym górnym wierzchołkiem byłby taki, który ma obie współrzędne ujemne, co wpłynęłoby na sam kształt algorytmu.

Należy również pamiętać, że $0.5 > -5 > -6$. Oznacza to, że nawet jeśli mielibyśmy punkty

$A(1,\ 10000)$ oraz $B(-100,\ 1)$ to wciąż $\frac{x_A}{y_A} > \frac{x_B}{y_B}$.

W zadaniu mamy również informację, że żadne dwa wierzchołki nie leżą w jednej linii. Przykładem takich wierzchołków byłyby: $A(-8, 4)$ oraz $B(-4, 2)$. W ich przypadku mielibyśmy $\frac{x_A}{y_A} = \frac{x_B}{y_B}$. Oczywiście nie musimy się tym zajmować, ponieważ zadanie zakłada, że takich wierzchołków nie ma.

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Załóżmy, że mamy wczytane dane do programu. Aby obliczyć $\frac{x_i}{y_i}$ dla i-tego wierzchołka wystarczy wykonać następującą operację:

```
1 X[i]/Y[i]
```

Warto tutaj zwrócić uwagę, że pomimo tego, że dzielimy liczby całkowite, to wynik będzie liczbą zmiennoprzecinkową. Rozwiązując zadanie za pomocą wybranego języka programowania, musimy o tym pamiętać, ponieważ jeżeli przypiszemy tę wartość do np. zmiennej typu `integer`, to algorytm nie zadziała poprawnie.

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Pamiętajmy również, że w naszych tablicach indeksy są numerowane od 1 do n zamiast od 0 do $n-1$ jak w językach C++, Python i Java. Jeżeli nie zwrócimy na to uwagi, to możemy uzyskać błędny wynik.

Optymalnym rozwiązaniem problemu byłoby przejście po wszystkich wierzchołkach i porównywanie, czy wynik operacji $\frac{x_i}{y_i}$ jest mniejszy od zapamiętanego wcześniej. Jeśli tak, powinniśmy zapamiętać ten wynik oraz jego pozycję w tablicy.

Podsumowując, optymalnym rozwiązaniem problemu będzie napisanie poprawnego algorytmu odnajdującego $\min\left(\frac{x_i}{yi}\right)$.

5

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

W specyfikacji podano, że są dwie tablice `X[0..n - 1]` oraz `Y[0..n - 1]`.

Oznacza to, że możemy założyć, że dane te są już wczytane do pamięci programu, podobnie jak wartość zmiennej n.

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Algorytm przyjąłby wstępnie następującą postać:

```
1 dla i = 0, 1, 2 ... do n -
  1 wykonuj:
2     jeżeli X[i]/Y[i] < ...
3
```

Właśnie mniejsze od czego? Wspomnieliśmy już, że optymalnym rozwiązaniem byłoby przejście po kolejnych wierzchołkach i porównanie, czy wynik operacji $\frac{x_i}{yi}$ jest mniejszy od zapamiętanego. Załóżmy więc, że na początku

zapamiętujemy wartość $\frac{x_i}{y_i}$ dla i=0 oraz `indeks=0`.

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Ostatecznie algorytm zapisany za pomocą pseudokodu wyglądałby następująco:

```
wczesniejszy = X[0]/Y[0]
indeks = 0
dla i = 0, 1, 2, 3 ..., n - 1 wykonuj:
    jeżeli X[i]/Y[i] < wczesniejszy:
        indeks = i
        wczesniejszy = X[i]/Y[i]
wypisz X[indeks]
wypisz Y[indeks]
```

8

Więcej zadań znajdziesz w zbiorze dostępnym na stronie internetowej Okręgowej Komisji Egzaminacyjnej w Warszawie.
Źródło: OKE, oke.waw.pl, tylko do użytku edukacyjnego.

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Przeanalizujmy schemat oceniania rozwiązania tego zadania:

Schemat punktowania

2 pkt – za poprawny algorytm, w tym:

1 pkt – za prawidłową inicjalizację oraz konstrukcję pętli,

1 pkt – za zastosowanie prawidłowego porównania oraz wyznaczenie współrzędnych skrajnie lewego szczytu.

Uwaga: za prawidłowe porównanie i wyznaczenie poprawnej najmniejszej wartości ilorazu współrzędnych oraz poprawnego indeksu – 1 pkt,

0 pkt – za podanie odpowiedzi błędnej albo brak odpowiedzi.

Przedstawione zasady dostępne są na stronie CKE.

Praca domowa

W wybranym języku programowania zaimplementuj algorytm z problemu 1.

```
1
```

```
1
```

Problem 1

Napisz za pomocą pseudokodu algorytm, który przestawi elementy tablic X i Y tak, aby szczyty były uporządkowane w kolejności, w której obserwator widzi je od lewej do prawej strony. Algorytm powinien mieć złożoność czasową kwadratową lub mniejszą.

Algorytm może używać wyłącznie instrukcji sterujących, operatorów arytmetycznych, operatorów logicznych, porównań i przypisań do zmiennych. Zabronione jest używanie funkcji bibliotecznych dostępnych w językach programowania.

Specyfikacja problemu:

Dane:

- n – liczba całkowita dodatnia
- `X[0..n - 1]` – tablica liczb całkowitych
- `Y[0..n - 1]` – tablica liczb całkowitych dodatnich
- Para `(X[i], Y[i])` to współrzędne jednego szczytu, `i = 0, 1, ..., n - 1`
- Żadne dwa szczyty nie leżą w jednej linii z obserwatorem.

Wynik:

- `X[0..n - 1], Y[0..n - 1]` – tablice zawierające współrzędne danych szczytów, uporządkowanych w kolejności, w której obserwator widzi je od lewej do prawej strony.

```
1
```

Polecenie 3

Porównaj swoje rozwiązanie z przedstawionym w prezentacji.

1

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

W poprzednim zadaniu wybieraliśmy najmniejszy element ze zbioru wszystkich $\frac{x_i}{yi}$. W tym mamy posortować ten zbiór rosnąco.

Pamiętajmy, że nasz algorytm powinien mieć złożoność kwadratową lub mniejszą. Oznacza to, że możemy użyć tutaj sortowania bąbelkowego lub sortowania przez wstawianie.

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Ponownie wczytamy do naszego algorytmu dane.

```
Wczytaj n
Wczytaj X[0..n - 1]
Wczytaj Y[0..n - 1]
```

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Następnie wybieramy algorytm sortowania. W tym przypadku użyjemy sortowania bąbelkowego, nic nie stoi jednak na przeszkodzie, żeby skorzystać z sortowania przez wstawianie czy przez wybór.

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Ogólny schemat sortowania bąbelkowego przyjmuje następującą postać:

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

5

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Utworzymy zatem zmienne pomocnicze x1 oraz y1. Posłużą nam do przechowywania danych na temat porównywanych szczytów.

```
Wczytaj n
Wczytaj X[0..n - 1]
Wczytaj Y[0..n - 1]
x1 = 0
y1 = 0
```

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Sortujemy szczyty. Nasz pseudokod ostatecznie przyjmie więc postać:

```
Wczytaj n
Wczytaj X[0..n - 1]
Wczytaj Y[0..n - 1]

dla i = 0, 1, 2 ... n - 1
wykonuj:
    dla j = 0, 1, 2 ... n
- 1 wykonuj:
        Jeżeli X[j]/Y[j] >
X[j + 1]/Y[j + 1]:
            x1 = X[j + 1]
            y1 = Y[j + 1]
            X[j + 1] =
X[j]
            Y[j + 1] =
Y[j]
            X[j] = x1
```

```
            Y[j] = y1

dla i = 0, 1, 2, 3 ... n
wykonuj:
        wypisz X[i]
        wypisz Y[i]

```

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Przeanalizujmy schemat oceniania:

Schemat punktowania

4 pkt – za poprawny algorytm, w tym:

1 pkt – za poprawną konstrukcję zewnętrznej pętli algorytmu sortowania,

1 pkt – za poprawną konstrukcję wewnętrznej pętli algorytmu sortowania,

1 pkt – za poprawne porównanie elementów,

1 pkt – za poprawną zamianę elementów uwzględniającą zarówno X, jak i Y.

Uwaga: za prawidłowe rozwiązanie o złożoności większej niż kwadratowa – maksymalnie 3

punkty,

0 pkt – za podanie odpowiedzi błędnej albo brak odpowiedzi.

8

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

W tym zadaniu w schemacie punktowania znajduje się następująca informacja:

Uwaga: za każde inne niż przedstawione niżej, ale całkowicie poprawne rozwiązanie, przyznajemy maksymalną liczbę punktów.

9

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Przykładowe rozwiązanie nr 1 (sortowanie bąbelkowe).

```
powtarzaj n - 1 razy:
    i = 0, 1, 2, ..., n - 1 wykonuj
        jeżeli X[i + 1]/Y[i + 1] < X[i]/Y[i]
            t = X[i]
            X[i] = X[i + 1]
            X[i + 1] = t
            t = Y[i]
            Y[i] = Y[i + 1]
            Y[i + 1] = t
```

10

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Przykładowe rozwiązanie nr 2 (sortowanie przez wybór).

```
dla i = 0, 1, 2, ..., n - 1 wykonuj
    m = i
```

```
    dla j = i + 1, i + 2, ..., n wykonuj
        jeżeli X[j] / Y[j] < X[m] / Y[m]
            m = j
    t = X[i]
    X[i] = X[m]
    X[m] = t
    t = Y[i]
    Y[i] = Y[m]
    Y[m] = t
```

11

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Przykładowe rozwiązanie nr 3 (sortowanie przez wstawianie).

```
dla i = 0, 1, 2, 3, ..., n - 1 wykonuj
    j = i
    dopóki j > 1 oraz X[j] / Y[j] < X[j - 1] / Y[j - 1]:
        t = X[j]
        X[j] = X[j - 1]
        X[j - 1] = t
        t = Y[j]
        Y[j] = Y[j - 1]
        Y[j - 1] = t
    j = j - 1
```

12

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/P1EKfaHQv

Pamiętajmy, że większość zadań można wykonać na wiele sposobów. Umiejętność zaimplementowania poszczególnych algorytmów przyda się w każdym z omawianych przypadków, dlatego tak ważne jest przyswojenie tej tematyki oraz wykonanie jak największej liczby zadań algorytmicznych.

Praca domowa

W wybranym języku programowania zaimplementuj algorytm z problemu 2.

```
1
```

```
1
```

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Dane są operacje logiczne na bitach `not`, `and` i `or` opisane poniżej:

a	**not** *a*
1	0
0	1

a	*b*	*a* **and** *b*
1	1	1
0	1	0
1	0	0
0	0	0

a	*b*	*a* or *b*
1	1	1
0	1	1
1	0	1
0	0	0

oraz wyrażenie $W(a, b)$:

$$((not\ a)\ and\ b)\ or\ (a\ and\ (not\ b))$$

Wyrażenie	Prawda	Fałsz
$W(0, 0) = 1$	P ◯	F ◯
$W(1, 0) = 1$	P ◯	F ◯
$W(0, 1) = 0$	P ◯	F ◯
$W(1, 1) = 0$	P ◯	F ◯
$W(1, 1) = 1$	P ◯	F ◯

> Zadanie pochodzi z I części arkusza maturalnego na poziomie rozszerzonym z czerwca 2021 roku. Cały arkusz znaleźć można na stronie Centralnej Komisji Egzaminacyjnej.

Ćwiczenie 2

Dwie różne liczby całkowite a i b większe od 1 nazwiemy skojarzonymi, jeśli suma wszystkich różnych dodatnich dzielników a mniejszych od a jest równa $b + 1$, a suma wszystkich różnych dodatnich dzielników b mniejszych od b jest równa $a + 1$.

Skojarzone są np. liczby 140 i 195, ponieważ:

- dzielnikami 140 są 1, 2, 4, 5, 7, 10, 14, 20, 28, 35, 70, a ich suma wynosi 196 = 195+1.
- dzielnikami 195 są 1, 3, 5, 13, 15, 39, 65, a suma tych liczb równa jest 141 = 140+1.

> Zadanie pochodzi z I części arkusza maturalnego na poziomie rozszerzonym z maja 2016 roku. Cały arkusz znaleźć można na stronie Centralnej Komisji Egzaminacyjnej.

Ćwiczenie 3

Zapoznaj się z algorytmem i jego specyfikacją i algorytmem, a następnie wykonaj ćwiczenie.

Dane:

- d – długość tekstu, liczba całkowita dodatnia
- s[1..d] – ciąg znaków o długości d

Wynik:
w – wartość logiczna

```
w = prawda;
dla i = 1, 2, ... do d/2 wykonuj:
    jeżeli s[i] != s[d-i+1]
        w = fałsz
zwróć w
```

Zastanów się, co sprawdza podany powyżej algorytm.

Ćwiczenie 4

Uzupełnij algorytm, tak żeby generowane przez niego wyniki zgadzały się z tabelą.

```
jeżeli n < 0:
    ....
w = 1
dla i = 1, 2, .. n wykonuj:
    ....
zwróć w
```

n	w
-2	0
5	120
0	1
3	6
1	1
2	2

n	w
-1	0

Zapisz uzupełniony algorytm.

```
1
```

Ćwiczenie 5

Zapoznaj się z specyfikacją i wykonaj ćwiczenie.

Specyfikacja:

Dane:

- n – liczba całkowita dodatnia

Wynik:

- w – liczba całkowita dodatnia

```
w = 1
a1 = 1
a2 = 1
jeżeli n == 1 lub n == 2:
    zwróć w
w przeciwnym wypadku dla i = 1, 2, ... n wykonuj:
    temp = a1 + a2
    a1 = a2
    a2 = temp
    w = a2
    zwróć w
```

Uzupełnij.

n = 1 | w =

n = 2 | w =

n = 3 | w =

n = 4 | w =

n = 5 | w =

n = 10 | w =

Poniższe zadanie zostało stworzone na podstawie zadania 3.1 z części pisemnej matury rozszerzonej z informatyki z roku 2021.

```
funkcja f(n):
    jeżeli n > 0
        wypisz n
        f(n – 7)
        wypisz n
```

Ćwiczenie 6

Zaznacz wszystkie zdania prawdziwe, na podstawie algorytmu podanego w ćwiczeniu 5.

- [ ] W wyniku wywołania $f(-1)$ otrzymamy ciąg pusty.
- [ ] W wyniku wywołania $f(7)$ otrzymamy ciąg $7\ 7$.
- [ ] W wyniku wywołania $f(22)$ otrzymamy ciąg $22\ 15\ 8\ 1\ 1\ 8\ 15\ 22$.
- [ ] W wyniku wywołania $f(0)$ otrzymamy ciąg $0\ 0$.
- [ ] W wyniku wywołania $f(8)$ otrzymamy ciąg $8\ 1\ 1\ 8$.
- [ ] W wyniku wywołania $f(1)$ otrzymamy ciąg $1\ 1$.
- [ ] W wyniku wywołania $f(22)$ otrzymamy ciąg $22\ 15\ 8\ 1\ 1\ 0\ 0\ 1\ 1\ 8\ 15\ 22$.

Ćwiczenie 7

```
funkcja FizzBuzz (n):
    ...
    ...
    ...
    w przeciwnym wypadku:
        zwróć n
```

Uzupełnij podany kod w ten sposób, żeby w przypadku otrzymania liczby podzielnej przez 3 program wypisywał słowo „Fizz”. W przypadku otrzymania liczby podzielnej przez 5 wypisywał „Buzz”, a w przypadku liczby podzielnej przez 15 niech wypisuje „FizzBuzz”. W przypadku, jeżeli liczba nie jest podzielna ani przez 3, ani przez 5, to ją zwróć.

```

```

Dla zainteresowanych

Podczas niektórych rozmów o pracę w sektorze programowania kandydat jest proszony o rozwiązanie takiego zadania, jak sformułowano w tym ćwiczeniu.

Poniżej **przykładowe** rozwiązanie:

Ćwiczenie 8

Korzystając z podanego algorytmu do obliczania średniej ważonej:

```
wczytaj N
wczytaj oceny[1...N]
wczytaj wagi[1...N]
suma = 0
sumaWag = 0
dla i = 1, 2 ... N:
    suma = suma + oceny[i] * wagi[i]
    sumaWag = sumaOcen + wagi[i]
zwróć suma / sumaWag
```

uzupełnij poniższą tabelę.

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Algorytmika – zadania maturalne

Grupa docelowa:

Szkoła podstawowa, szkoła ponadpodstawowa, liceum ogólnokształcące, technikum, zakres 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.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres podstawowy. Uczeń:

1) planuje kolejne kroki rozwiązywania problemu, z uwzględnieniem podstawowych etapów myślenia komputacyjnego (określenie problemu, definicja modeli i pojęć, znalezienie rozwiązania, zaprogramowanie i testowanie rozwiązania).

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 strukturę egzaminu maturalnego z informatyki.
- Omówisz typy zadań teoretycznych występujących na egzaminie maturalnym z informatyki.
- Rozwiążesz zadania z arkuszy maturalnych dotyczące algorytmów.

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 w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- e-podręcznik;
- 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: „Algorytmika – zadania maturalne". Nauczyciel prosi uczniów o zapoznanie się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla uczniom temat, wskazuje cele zajęć oraz ustala z uczestnikami zajęć kryteria sukcesu.

Faza realizacyjna:

1. **Praca z tekstem.** Nauczyciel wyświetla zawartość sekcji „Przeczytaj". Na forum klasy uczniowie analizują przedstawione w niej rozwiązania Przykładów 1-6.

2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Prezentacja multimedialna". Uczniowie wspólnie zapoznają się z treścią Polecenia 1. Następnie samodzielnie piszą algorytm za pomocą pseudokodu, który znajdzie i poda współrzędne skrajnie lewego szczytu, tzn. widocznego dla obserwatora na lewo od wszystkich pozostałych szczytów. Następnie porównują swoje rozwiązanie z przedstawionym w prezentacji. W kolejnym kroku uczniowie zapoznają się z Poleceniem 2. W parach wypracowują rozwiązanie, które następnie porównują z prezentacją.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują ćwiczenia nr 1-8 z sekcji „Sprawdź się". Nauczyciel sprawdza poprawność wykonanych zadań, omawiając je wraz z uczniami.

Faza podsumowująca:

1. Nauczyciel ponownie wyświetla na tablicy temat i cele lekcji zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji następuje omówienie ewentualnych problemów z rozwiązaniem ćwiczeń z sekcji „Sprawdź się".
2. Na koniec zajęć nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak…".

Praca domowa:

1. Uczniowie wykonują pracę domową z sekcji „Prezentacja multimedialna".

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:

- Multimedia w sekcjach: „Przeczytaj", „Prezentacja multimedialna", „Sprawdź się" można wykorzystać jako materiał służący powtórzeniu materiału.