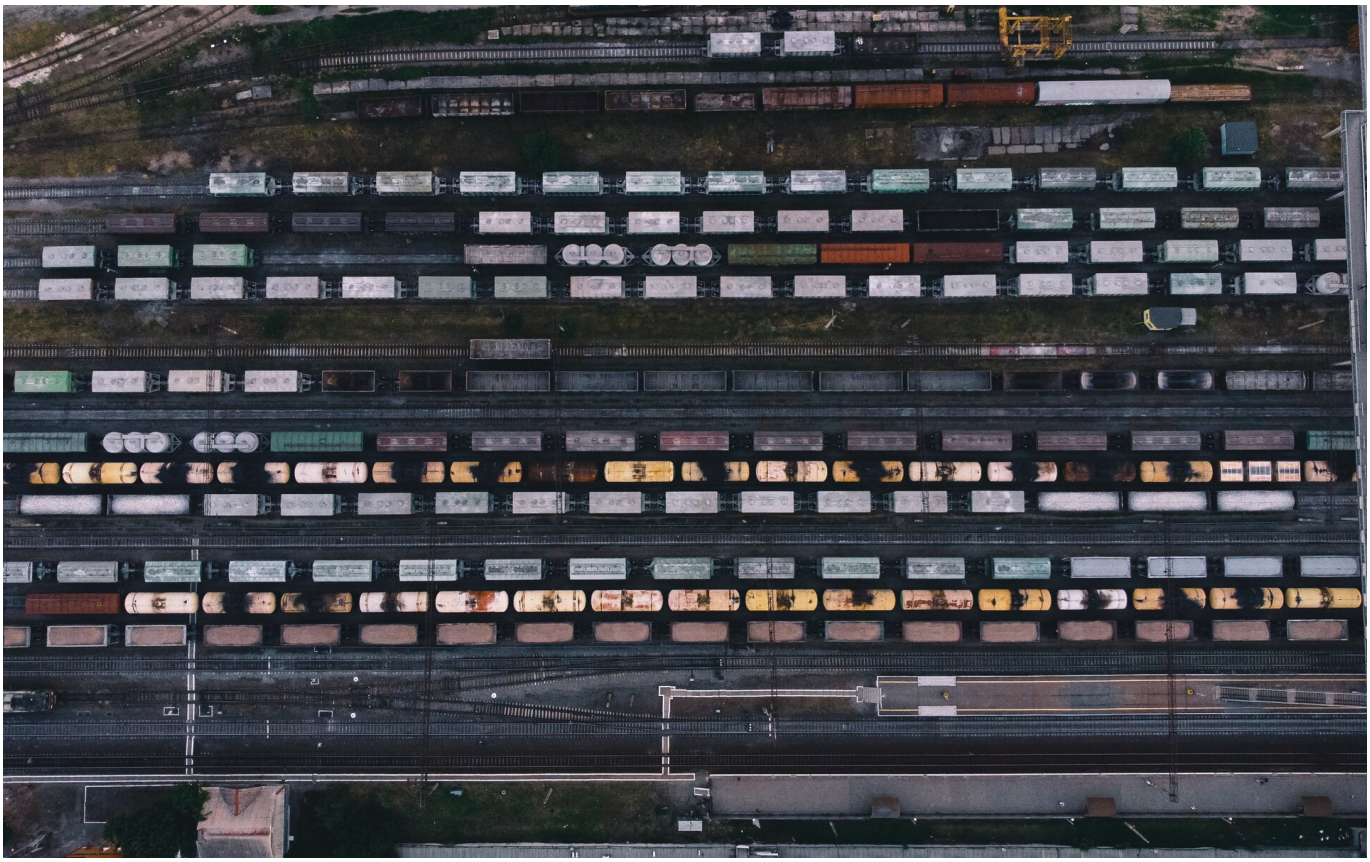


Tablice wielowymiarowe w języku C++

Źródło: Max Vertsanov, domena publiczna.

> W tym e-materiale powtarzamy wiadomości ze szkoły podstawowej.

W e-materiale Tablice wielowymiarowe poznaliśmy definicję tego typu tablic. Czy potrafisz podać ich przykłady z życia codziennego?

Jeśli nie, przypomnij sobie popularną grę w statki. Plansza jest niczym innym jak przykładem tablicy wielowymiarowej, a dokładnie tablicy dwuwymiarowej. Składa się ze 100 pól, każde jest oznaczone przez jedną literę oraz jedną liczbę i przechowuje pewną informację, w tym przypadku o obecności lub nieobecności statku.

W tym e-materiale zapoznamy się z implementacją tablic wielowymiarowych w języku C++.

Implementację tablic wielowymiarowych w wybranych językach programowania znajdziesz w e-materiałach:

- Tablice wielowymiarowe w języku Java,
- Tablice wielowymiarowe w języku Python.

Więcej zadań? Przejdź do e-materiału Tablice wielowymiarowe – zadania maturalne.

Informacje na temat tablic jednowymiarowych znajdziesz w e-materiałach:

- Tablice jednowymiarowe,
- Tablice jednowymiarowe w języku C++.

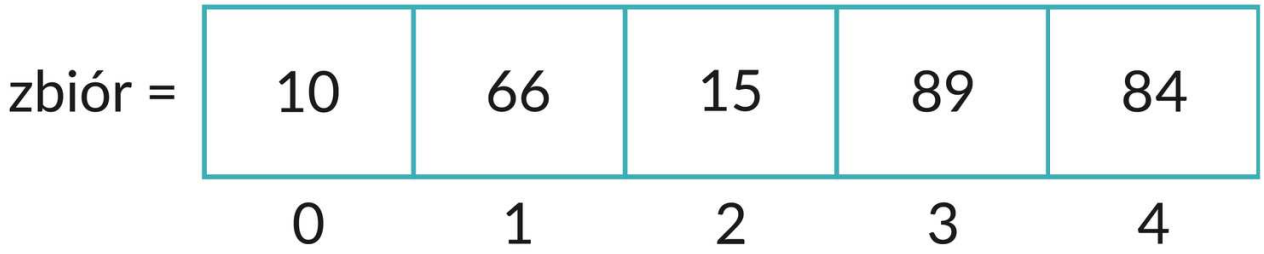
Twoje cele

- Scharakteryzujesz tablicę dwuwymiarową.
- Omówisz definicję i zastosowania macierzy.
- Zaimplementujesz program, który wykorzystuje macierze.
- Rozwiążesz zadania, wykorzystując tablice dwuwymiarowe.

Przeczytaj

Definicja i własności tablic dwuwymiarowych

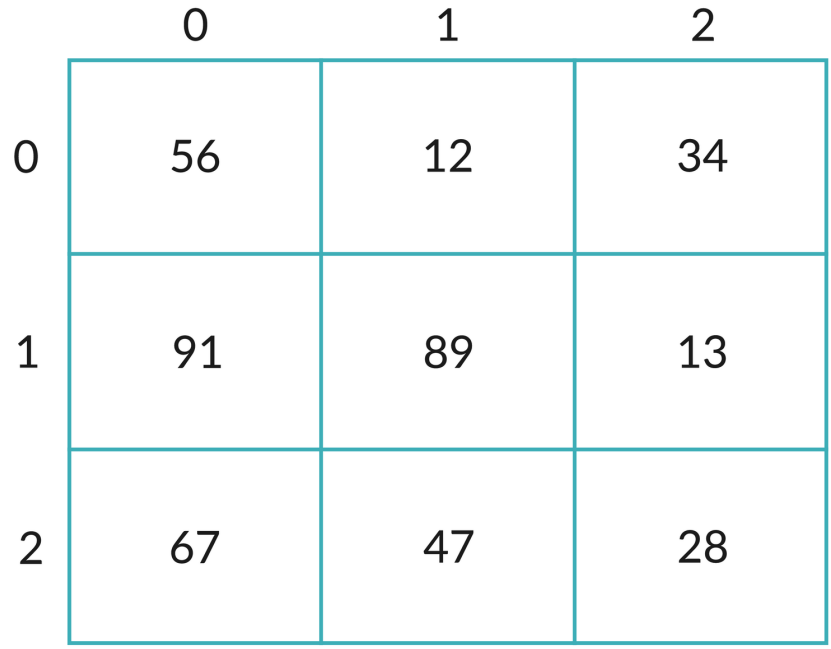
Zazwyczaj o tablicy myślimy jak o ciągu komórek, w których możemy przechowywać wartości tego samego typu. Każda z komórek ma swój unikatowy numer – indeks. Jeżeli zwizualizujemy tablicę, to może ona wyglądać tak:



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

Powyższa tablica `zbior` ma długość równą 5. Możemy ją skojarzyć z listą elementów. Jest to tablica jednowymiarowa.

Tablica dwuwymiarowa to inaczej jedna tablica, która zawiera w sobie inne tablice, a te z kolei zawierają wartości. W przedstawionym przykładzie każdy z elementów tablicy ma dwa indeksy.



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

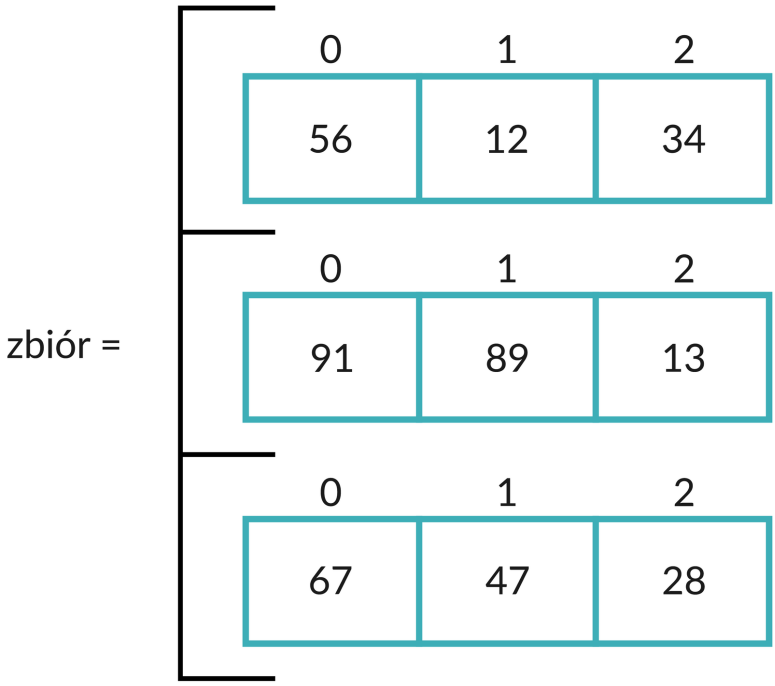
W tym przypadku liczba 89 znajduje się na miejscu o indeksach 1, 1. Podobnie jak w przypadku tablicy jednowymiarowej, aby dostać się do wybranego elementu, podajemy jego indeks w nawiasach kwadratowych. W tablicy dwuwymiarowej podajemy dwa indeksy, pierwszym z nich jest indeks wiersza, a kolejnym jest indeks kolumny. Jeżeli chcemy

wypisać wartość zapisaną w wierszu o indeksie 2 i kolumnie o indeksie 1, w języku C++ używamy następującej linijki kodu:

```
cout << zbior[2][1] << endl;
```

Wynikiem działania tej instrukcji będzie wypisanie liczby 47 w konsoli.

Tablicę dwuwymiarową możemy przyrównać do tablicy tablic wartości danego typu. Jeżeli spróbujemy uzyskać wartość z tablicy dwuwymiarowej, podając tylko jeden indeks, uzyskamy tablicę jednowymiarową. Prezentowanie tablicy dwuwymiarowej przez tabelę jest wygodniejsze do opisania sposobu działania programu. Z kolei w celu zrozumienia sposobu działania samych tablic dwuwymiarowych użyjemy poniższej wizualizacji.



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

Jak widzimy powyżej, mamy trzy tablice, które przechowują wartości liczbowe. Te trzy tablice są przechowywane w tablicy o nazwie `zbior`. Dlatego też wywołując:

```
zbior[2]
```

jako wynik otrzymujemy tablicę, która w tablicy `zbior` ma indeks o wartości 2. Następnie z otrzymanej tablicy pobieramy element o indeksie 1. Te dwa wywołania łączymy w jedno, zapisując indeksy obok siebie:

```
zbior[2][1]
// zbior[2] -> tablica
```

```
3 // tablica[1] -> wartość 47
```

Znamy już strukturę danych, jaką są tablice dwuwymiarowe, ale nadal nie wiemy, jak je zdefiniować. W języku C++ jest to bardzo podobne do tworzenia tablicy jednowymiarowej.

```
1 int zbior[3][3];
```

Na początek określamy typ elementów tablicy. W tym przypadku jest to `int`. Następnie podajemy nazwę tablicy oraz liczbę wierszy i kolumn. Tablica `zbior` będzie miała trzy wiersze oraz trzy kolumny, dokładnie tak jak w przykładzie.

Po takiej inicjalizacji tablica `zbior` będzie wypełniona wartościami znajdującymi się w obszarze pamięci, w którym tablica została zapisana. Jeżeli chcemy, aby tablica została zainicjowana podanymi przez nas liczbami, używamy tego zapisu:

```
1 int zbior[3][3] = {{56, 12, 34}, {91, 89, 13}, {67, 47, 28}};
```

Pierwsza klamra reprezentuje tablicę `zbior`, natomiast klamry zawarte w niej reprezentują tablice, które są jej elementami. Jeżeli w tym zapisie nie podamy wszystkich elementów, te, które nie zostaną podane, będą zainicjowane wartością 0.

> **Ciekawostka**
>
> Jeżeli nie podamy wartości dla żadnego elementu, cała tablica zostanie zainicjowana zerami. Jedyne co musimy zrobić to:
>
> ```
> 1 int zbior[3][3] = {};
> ```

Ponieważ tablice dwuwymiarowe mają dwa indeksy, przejście przez wszystkie pola będzie nieco trudniejsze. Dobrą praktyką jest tworzenie zmiennych, które będą przechowywać rozmiar tablicy. W przypadku tablicy dwuwymiarowej będą potrzebne dwie takie zmienne – jedna będzie przechowywać liczbę wierszy, natomiast druga liczbę kolumn. Do przeglądania tablicy dwuwymiarowej najwygodniej będzie zastosować dwie pętle `for`, chociaż można użyć dowolnych pętli. Pierwsza z nich posłuży do przejścia przez wiersze, a druga przez kolumny.

Zobaczmy, jak będzie wyglądać prosty program wypisujący wszystkie wartości z tablicy do konsoli.

```
1 for (int i = 0; i < liczbaWierszy; i++) {
```

```
    for (int j = 0; j < liczbaKolumn; j++) {
        cout << zbior[i][j] << " ";
    }

    cout << endl;
}
```

Tablice wielowymiarowe

Tak jak w przypadku tablic jednowymiarowych, elementami tablic dwuwymiarowych mogą być także inne typy danych niż liczby. Mogą to być na przykład wartości tekstowe, logiczne, czy zdefiniowane przez użytkownika struktury danych.

Szczególna sytuacja ma miejsce wtedy, kiedy elementem tablicy dwuwymiarowej jest kolejna tablica jednowymiarowa. W ten sposób powstaje tablica trójwymiarowa. Rządzi się ona dokładnie takimi samymi prawami jak tablica dwuwymiarowa – oczywiście z tym wyjątkiem, że musimy podawać dodatkowy, trzeci indeks.

Przykładowa tablica trójwymiarowa, wraz z kodem wyświetlającym wszystkie jej elementy:

```
    int trojwymiar[3][3][3] =
    {
        {{1,1,1}, {2,2,2}, {3,3,3}},
        {{4,4,4}, {5,5,5}, {6,6,6}},
        {{7,7,7}, {8,8,8}, {9,9,9}}
    };

    for(int i = 0; i < 3; i++)
        for(int j = 0; j < 3; j++)
            for(int k = 0; k < 3; k++)
                cout << trojwymiar[i][j][k];
```

> **Ciekawostka**
>
> Tablice wielowymiarowe (najczęściej dwu- lub trójwymiarowe) są powszechnie używane do reprezentacji światów w grach komputerowych. Popularna gra *Minecraft* do przechowywania bloków w poszczególnych częściach świata używa właśnie tablic trójwymiarowych. Same części świata, tak zwane *chunki*, przechowywane są z kolei w tablicy dwuwymiarowej.

Nic nie stoi na przeszkodzie, abyśmy w ten sam sposób tworzyli tablice o większej liczbie wymiarów. Tablicę czterowymiarową stworzymy, implementując tablicę trójwymiarową,

której elementami są tablice jednowymiarowe.

Słownik

> **macierz**
>
> zbiór liczb lub wyrażeń zapisanych w postaci prostokątnej tablicy
> **tablica**
>
> kontener (struktura danych) służący do przechowywania danych (wartości) tego samego typu; każdy element ma określony indeks (kolejny numer); w pamięci komputera elementy tablicy są ułożone kolejno jeden obok drugiego

Film samouczek

Polecenie 1

Przeanalizuj prezentację, następnie wykonaj polecenia.

Polecenie 2

Napisz program, który odejmie od siebie dwie wypełnione liczbami rzeczywistymi macierze o `m` kolumnach i n wierszach i wyświetli wynik tego działania w postaci macierzy wynikowej. Przetestuj program dla dwóch macierzy o wymiarach 4 x 4.

Specyfikacja problemu:

Dane:

- n – liczba naturalna dodatnia; liczba wierszy
- `m` – liczba naturalna dodatnia; liczba kolumn
- `macierzA` – macierz o `m` kolumnach i n wierszach; tablica liczb rzeczywistych
- `macierzB` – macierz o `m` kolumnach i n wierszach; tablica liczb rzeczywistych

Wynik:

- macierz wynikowa działania `macierzA - macierzB`; tablica liczb rzeczywistych

Polecenie 3
Napisz program, który doda do siebie dwie wypełnione liczbami rzeczywistymi macierze o `m` kolumnach i `n` wierszach i wyświetli wynik tego działania w postaci macierzy wynikowej. Przetestuj działanie programu dla tablicy o wymiarach 3 x 4.

Specyfikacja problemu:

Dane:

- `n` – liczba naturalna dodatnia; liczba wierszy
- `m` – liczba naturalna dodatnia; liczba kolumn
- `macierzA` – macierz o `m` kolumnach i `n` wierszach; tablica liczb rzeczywistych
- `macierzB` – macierz o `m` kolumnach i `n` wierszach; tablica liczb rzeczywistych

Wynik:

- macierz wynikowa działania `macierzA + macierzB`; tablica liczb rzeczywistych

Polecenie 4
Porównaj swoje rozwiązanie z filmem.

Trwa wczytywanie danych ..

Film dostępny pod adresem /preview/resource/Rpp70SExyc8Ig

Film nawiązujący do treści materiału: Tablice dwuwymiarowe.

Plik o rozmiarze 606.00 B w języku polskim

Plik o rozmiarze 1.04 KB w języku polskim

Sprawdź się

Pokaż ćwiczenia:

Ćwiczenie 1

Napisz program, który do każdego elementu tablicy dodaje 10, a następnie wypisze jej zawartość tak, żeby poszczególne wartości były oddzielone pojedynczym znakiem odstępu. Swój program przetestuj dla następującej tablicy:

```
tablica[n][m] = {{543, 654, 432, 154}, {546, 765, 875, 532},
```

Specyfikacja problemu:

Dane:

- `tablica` – tablica o wymiarach `n x m`; elementy tablicy to liczby całkowite
- `n` – liczba naturalna dodatnia, liczba wierszy
- `m` – liczba naturalna dodatnia; liczba kolumn

Wynik:

Na standardowym wyjściu wyświetlane są wierszami (wszystkie w jednej linii, oddzielone od siebie pojedynczym znakiem odstępu) wartości wszystkich komórek tablicy `tablica`.

Ćwiczenie 2

Napisz program, który sprawdzi, czy dwie tablice zawierają te same wartości w identycznej kolejności. Program powinien wypisywać komunikat TAK, jeśli tablice zawierają te same wartości, lub NIE w przeciwnym przypadku Swój program przetestuj dla następujących tablic:

```
tablicaA[n][n] = {{543, 546, 764}, {235, 765, 976}, {123, 543,
tablicaB[n][n] = {{543, 546, 764}, {235, 765, 976}, {123, 553,
```

Specyfikacja problemu:

Dane:

- `tablicaA` – tablica o wymiarach `n x m`; elementy tablicy to liczby całkowite
- `tablicaB` – tablica o wymiarach `n x m`; elementy tablicy to liczby całkowite
- `n` – liczba naturalna dodatnia; liczba wierszy
- `m` – liczba naturalna dodatnia; liczba kolumn

Wynik:

Na standardowym wyjściu wyświetlany jest komunikat TAK, jeśli tablice zawierają te same wartości, lub NIE w przeciwnym przypadku.

Ćwiczenie 3

Napisz program, który wypełni tablicę `macierzWynik` w opisany sposób, a następnie ją wypisze.

Dane są dwie tablice: `macierzA, macierzB` o wymiarach `n x m`. Tablice reprezentują macierze. Chcemy utworzyć trzecią macierz `macierzWynik` o takich samych wymiarach. Zostanie ona wypełniona w ten sposób, że dla każdych $i \in \langle 0, n \rangle, j \in \langle 0, m \rangle$ zachodzi równość:

$$macierzWynik[i][j] = \max(macierzA[i][j], macierzB[i][j])$$

Swój program przetestuj dla następujących macierzy:

```
macierzA[n][m] = {{346, 654, 865}, {243, 765, 869}, {543, 758,
macierzB[n][m] = {{123, 765, 867}, {356, 543, 235}, {649, 535,
```

Wynik dla podanych macierzy:

```
346 765 867
356 765 869
649 758 865
```

Specyfikacja problemu:

Dane:

- `macierzA` – tablica o wymiarach `n x m`; elementy tablicy to liczby całkowite
- `macierzB` – tablica o wymiarach `n x m`; elementy tablicy to liczby całkowite
- `n` – liczba naturalna dodatnia; liczba wierszy
- `m` – liczba naturalna dodatnia; liczba kolumn

Wynik:

> Na standardowym wyjściu wyświetlana jest zawartość tablicy `macierzWynik`, utworzonej w opisany wyżej sposób (każdy wiersz macierzy w osobnej linii, wartości w wierszu oddzielone pojedynczym znakiem odstępu).

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Tablice wielowymiarowe w języku C++

Grupa docelowa:

Liceum ogólnokształcące i technikum, 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.

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

2) do realizacji rozwiązania problemu dobiera odpowiednią metodę lub technikę algorytmiczną i struktury danych;

3) objaśnia dobrany algorytm, uzasadnia poprawność rozwiązania na wybranych przykładach danych i ocenia jego efektywność;

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

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;

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):

- Scharakteryzujesz tablicę dwuwymiarową.
- Omówisz definicję i zastosowania macierzy.
- Zaimplementujesz program, który wykorzystuje macierze.
- Rozwiążesz zadania, wykorzystując tablice dwuwymiarowe.

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:

- 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;
- komputery z głośnikami, słuchawkami i dostępem do internetu lub telefony z dostępem do internetu;
- zasoby multimedialne zawarte w e-materiale;

- oprogramowanie dla języka C++ przeznaczone na urządzenia mobilne (np. CCxxdroid 4.0).

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Tablice wielowymiarowe w języku C++". Uczniowie mają zapoznać się z treściami w sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel inicjuje rozmowę wprowadzającą w temat lekcji. Przedstawia cele zajęć oraz kryteria sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel prosi wybranego ucznia lub uczniów o przedstawienie sytuacji problemowej związanej z tematem lekcji oraz w odniesieniu do poznanych języków programowania.

Faza realizacyjna:

1. **Praca z tekstem.** Jeżeli przygotowanie uczniów do lekcji jest niewystarczające, nauczyciel prosi o indywidualne zapoznanie się z treścią zawartą w sekcji „Przeczytaj". Każdy uczestnik zajęć podczas cichego czytania wynotowuje najważniejsze kwestie poruszane w tekście.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Film samouczek". Uczniowie zapoznają się z prezentacją.
   Następnie w parach wykonują polecenia 2 i 3, a w kolejnym kroku porównują swoje rozwiązania z przedstawionym w filmie.
3. **Ćwiczenie umiejętności.** Prowadzący zapowiada uczniom, że w kolejnym kroku będą rozwiązywać ćwiczenia nr 1-2 z sekcji „Sprawdź się". Każdy z uczniów robi to samodzielnie. Po ustalonym czasie wybrani uczniowie przedstawiają rozwiązania. Nauczyciel w razie potrzeby koryguje odpowiedzi, dopowiada istotne informacje, udziela uczniom informacji zwrotnej.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia, zwracając uwagę na nabyte umiejętności, omawia ewentualne problemy podczas rozwiązania ćwiczeń z programowania w języku C++.

Praca domowa:

1. Uczniowie wykonują ćwiczenie 3 z sekcji „Sprawdź się".

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.

Wskazówki metodyczne:

- Treści w sekcji „Film samouczek” można wykorzystać jako materiał służący powtórzeniu materiału.