Tablice

#array #index #vararg #ArrayIndexOutOfBounds #multidimension

Author: Piotr Niemczuk
Nickname: Pyoneru

Teoria

Wyobraź sobie że jesteś młodym człowiekiem (bo jesteś), pragnącym odpłynąć w fantastyczne światy. Nabywasz swoją pierwszą książkę, *Pan Lodowego Ogrodu, Jarosław Grzędowicz*. Jesteś nią zafascynowany i przeznaczasz dla niej specjalne miejsce na półce, gablotce, itd..

Po jej przeczytaniu sięgasz po część następną i ponownie przeznaczasz dla niej specjalne miejsce, może nawet obok tomu pierwszego.

Mijają lata i gromadzisz swoją biblioteczkę. Jednak nie masz na tyle miejsca aby każda z nich miała wydzielone miejsce. Specjalne, wyodrębione miejsce, rzucające się w oczy zamieniło się w ściany z indexami pozwalającymi szybko odnaleźć tytuł.

W bibliotece znajdziesz działy książek np. Nauki Ścisłe, Bibliografia, Historia czy Literatura, które następnie dzielisz na indeksy, np. alfabetycznie według nazwisk autorów lub tytułów.

W programowaniu również rozwiązujemy podobny problem.

Wyobraź sobie że tworzysz grę w której poruszasz pojedynczą postacią.

SP 10 woja btw.

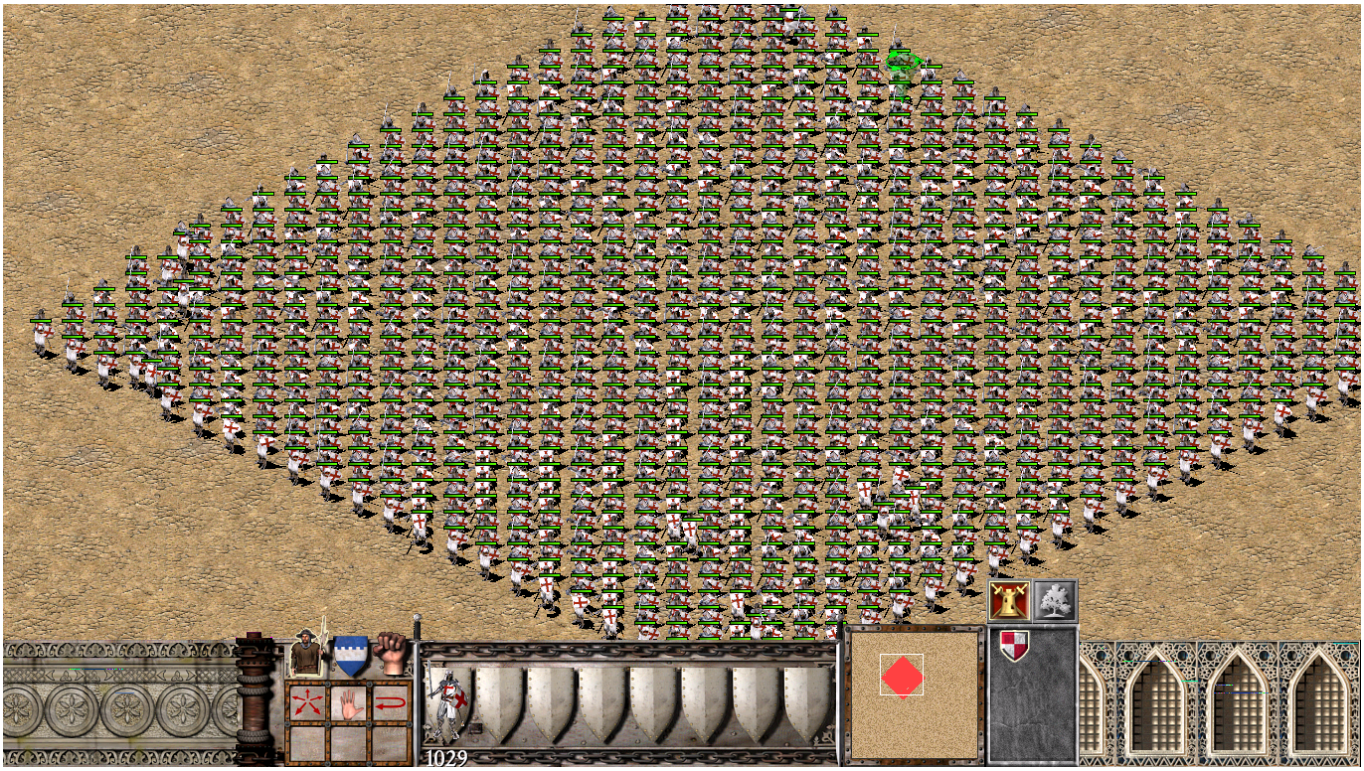
Potrzebujesz przechować o niej informacje.

- Pozycja w grze
- Nazwa postaci
- Tytuł
- HP (Punkty Życia)

Nie trudno przechować tyle informacji w zmiennych.

```
int pos_x;
int pos_y;
String nickname;
String title;
int hp;
```

Ale co gdy w grze nie operujemy nawet na jednej, dwóch czy trzech postaciach, a na setkach czy tysiącach? Jak teraz przechować tyle informacji w naszym programie?



Pozycja i HP dla 1029 jednostek.. sporo tych zmiennych by było, `pos_x`, `pos_y` i `hp` razy 1029 jednostek.. sporo tego by było. A potem jeszcze to spamiętaj!

Dlatego w programowaniu powstało coś na wzór bibloteki dla zmiennych, czyli tablice. Przechowują one nam zbiór podobnych danych.

Jak by to wyglądało w naszych pojemnikach?

Jeżeli chciałbyś przechować, nie pojedynczą zmienną, a tablice zmiennych, to powiadamiasz pracowników że potrzebujesz tablice o jakimś rozmiarze, np. 1029 dla naszych rycerzy!

Pracownicy magazynu, przygotowują pojemniki **jeden koło drugiego.** A następnie dostajesz klucz do tej tablicy.

Gdy chcesz odnieść się do zawartości z pojemnika, musisz pracowniką powiedzieć numer pojemnika (jego index.)

> ⓘ **Uporządkowany zbiór danych.**
>
> Tworząc tablice masz zawsze pewność że pojemniki na te dane będą koło siebie, dzięki czemu masz szybki dostęp między sąsiadującymi pojemnikami.

> ⓘ **Index**
>
> Numer pojemnika w tablicy, nazywamy indexem.
> Np. Element tablicy o indexie 0.

> ⚠ **Jak liczą programiści? 1,2,3.. NIE!**
>
> Jest to coś, co każdy programista musi się nauczyć. Od dziś nie liczysz od 1, tylko od 0! Tak! Jeżeli pracowniką magazynu powiesz że chcesz pojemnik o indexie 1, to tak na prawdę dostaniesz drugi w kolejności pojemnik!
> Dlatego pamiętaj że gdy stworzysz tablice o rozmiarze 10, to pierwszy jej element ma index 0, a ostatnii 9.

Praktyka

Jak tworzymy tablice?

```
String[] names = new String[10];
```

Aby zrobić z zmiennej danego typu, tablice tego typu to dodamy do niej `[]`.
Informujemy tym że jest to klucz, nie do pojedynczego pojemnika, a całej grupy pojemników;

```
String name; // pojedynczy pojemnik
String[] names; // grupa pojemników
```

Nazewnictwo

Jeżeli pracujemy na tablicach, to nazywamy je w liczbie mnogiej, np. `names` zamiast `name`. Dzięki tej prostej zasadzie, wiemy już po samej nazwie zmiennej że mamy do czynienia z tablicą, a nie pojedynczą zmienną.

Klucz nie jest jeszcze pojemnikami.

Sam zapis

```
String[] names;
```

mówi że zmienna `names` jest kluczem, do grupy pojemników typu `String`. Jednak sam klucz nie jest jeszcze przypisany do żadnej konkretnej grupy.

Mamy dwie możliwości aby przypisać do niego grupę pojemników.

1: Stworzyć nową

```
names = new String[10];
```

2: Przypisać do już istniejącej.

```
String[] cats = new String[10];
names = cats;
```

Dostęp do elemntu tablicy (pojemnika)

Aby dostać się do elementu tablicy(*ang. array*), przypisać do niego wartość lub odczytać, użyjemy tych samych nawiasów kwadratowych `[]`.

Przypisanie wartości.

```
names[0] = "Marta"; // pierwszy element tablicy
names[9] = "Kasia"; // ostatnii element tablicy.
```

Odczytanie wartości.

```
System.out.println(names[0]); // Marta
```

Żeby odnieść się do elementu tablicy, musimy podać jego index, indexem jest stała **dosłowna typu int**.

> ⓘ **A czy może być stała dosłowna typu long?**
>
> Nie, nie możemy tworzyć tak dużych tablic.

ArrayIndexOutOfBoundsException

Jeżeli spróbujemy się odnieść do nie istniejącego elementu tablicy to dostaniemy taki błąd.

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException Create breakpoint : Index 7 out of bounds for length 5
    at Main.main(Main.java:17)
```

W tym przypadku próbuje odnieść się do elementu o indexie 7, kiedy tablica ma rozmiar 5. Czyli jej ostatnim elementem jest index 4.

Rozmiar tablicy

Tablice dostarczają nam dodatkowych właściwości. Mogą nam zdradzić jakiego są rozmiaru.

```
String[] names = new String[10];
System.out.println(names.length);
```

Może się nam to przydać kiedy będziemy przekazywać tablice do funkcji, ponieważ w definicji tablicy mówimy tylko że funkcja przyjmie tablice, ale nie wiemy jakiego rozmiaru.

```
public static void printAll(String[] names){
        for(int i = 0; i < names.length; i++)
                System.out.println(names[i]);
}
```

Inicjalizacja tablicy

Tablice możemy tworzyć w sposób klasyczny

```
String[] roles = new String[10];
roles[0] = "senpai";
roles[1] = "pupil";
roles[2] = "spy";
roles[3] = "GameDev";
roles[4] = "gierki";
roles[5] = "WorkingOnCourse";
roles[6] = "filozof-się-znalazł"
```

Ale mamy możliwość wygodniejszego za-inicjalizowania tablicy podczas jej tworzenia.

```
String[] roles = new String[]{
        "senpai", "pupil", "spy", "GameDev", "gierki",
        "WorkingOnCourse", "filozof-się-znalazł"
}; // Pamiętaj o średniku!
```

Po nawiasach `[]` możemy umieścić nawiasy `{}` i po przecinku podawać kolejne wartości.

Rozmiar tablicy będzie równy liczbie elementów jakie podaliśmy w ten sposób.

Wartości domyślne

Co jeżeli spróbujemy odczytać wartość z tablicy przed jej przypisaniem? Spróbuj, wypróbuj różne typy!

Czy możemy zmienić wielkość tablicy?

Niestety nie możemy zmienić wielkości, raz już stworzonej tablicy. Możemy natomiast stworzyć nową tablice, większą i przenieść wartości z mniejszej.

```
String[] small = new String[] {"senpai", "pupil", "spy"}; // rozmiar = 3
String[] big = new String[15];

for(int idx = 0; idx < small.length; idx++){
        big[idx] = small[idx]; // Odczytaj wartość z tablicy small i przypisz
ją do tablicy big.
}
```

> **Struktury danych**
>
> Istnieje sposób aby dynamicznie zmieniać wielkość tablicy, taki problem rozwiązuje jedna z struktur danych, ale nie poznamy jej dzisiaj. Przepracuj wpierw tablice

Zmienna jako rozmiar tablicy.

Zamiast stałej dosłownej

```
String[] names = new String[10];
```

mozemy użyć zmiennej do zdefiniowania wielkośći tablicy.

```
Scanner in = new Scanner(System.in);
int size = in.nextInt();

String[] names = new String[size];
```

Tablice wielo wymiarowe

Możemy tworzyć również tablice wielo wymiarowe(*ang. multi dimension*).

```
int[][] chessBoard = new int[8][8];
```

Zasada działania pozostaje ta sama co przy tablicach jedno wymiarowych, z tym że teraz mamy więcej nawiasów :)

Jak zainicjalizować tablice wielowymiarowe? O, tak

```
int[][] chessBoard = new int[][]{
        {1, 2, 3, 5, 6, 7, 8},
        {1, 2, 3, 5, 6, 7, 8},
        {1, 2, 3, 5, 6, 7, 8},
        {1, 2, 3, 5, 6, 7, 8},
        {1, 2, 3, 5, 6, 7, 8},
        {1, 2, 3, 5, 6, 7, 8},
```

```
        {1, 2, 3, 5, 6, 7, 8},
};
```

Używamy kolejnych `{}` aby zdefiniować kolejny wiersz :)

Tip na koniec - pointer

Aby stworzyć tablice musimy podać jej rozmiar. Jeżeli chcielibyśmy napisać program który m.in. będzie przechowywał nazwy użytkowników to musimy sobie wyznaczyć maksymalną ich liczbe.

```
int max = 1_000_000;
String[] nicknames = String[max];
```

Następnie przy dodawaniu kolejnego użytkownika, musimy w któreś miejsce przypisać jego nazwe.

Tylko w które?

Pomoże nam w tym kolejna zmienna

```
int pointer = 0;
```

Przechowa kolejny wolny slot, zaczynamy od 0 czyli od początku.

Teraz gdy dodamy nowgo użytkownika, zwiększamy także wartość zmiennej `pointer`.

```
nicknames[pointer] = name;
pointer++;
```

W ten sposób wiemy w które miejsce dodać nowego użytkownika :)

Podsumowanie

To był ostatnii materiał... z pierwszego etapu! Teraz czeka Cię utrwalanie wszystkiego czego się nauczyłeś/aś do tej pory.
Przed kolejnym etapem poznasz jeszcze pare narzędzi które dostarcza nam język Java.
Napiszesz również coś większego i ciekawszego!

Good luck!