Wyjątki:

Wyjątki w języku Java są specjalnymi klasami, które służą do sygnalizowania, że w trakcie działania programu wystąpił problem. Wyjątki pozwalają programowi reagować na te sytuacje i podejmować odpowiednie działania. W Javie wyjątki dzielimy na dwie główne kategorie: wyjątki kontrolowane (checked exceptions) i wyjątki niekontrolowane (unchecked exceptions).

1. Wyjątki kontrolowane (checked exceptions):

Wyjątki kontrolowane to te, które muszą być obsłużone lub zgłoszone w deklaracji metody za pomocą słowa kluczowego `throws`. Należy to zrobić wtedy, gdy kompilator wskazuje, że dana metoda może rzucać tego rodzaju wyjątki.

Przykładem wyjątków kontrolowanych są:
IOException,
SQLException,
ClassNotFoundException itp.

Wyjątki kontrolowane wynikają często z operacji, które mogą być podatne na błędy, takie jak operacje na plikach, bazach danych lub sieci.

2. Wyjątki niekontrolowane (unchecked exceptions):

- Wyjątki niekontrolowane to te, które nie muszą być obsługiwane ani deklarowane w sygnaturze metody. Są one rzucane w wyniku błędów programistycznych, takich jak dzielenie przez zero (ArithmeticException) lub dostęp do indeksu poza zakresem tablicy (ArrayIndexOutOfBoundsException).
- Wyjątki niekontrolowane dziedziczą po klasie RuntimeException lub jej podklasach.

3. Error

Wyjątki związane z JVM, takie jak OutOfMemoryError czy StackOverflowError, są wyjątkami, na które programista ma niewielki wpływ, ponieważ wynikają one z problemów związanych z działaniem samej maszyny wirtualnej Java.

Klasy obsługi wyjątków w Javie dziedziczą z klasy `java.lang.Exception`. Klasy te zawierają metody i konstruktory, które pozwalają na zdefiniowanie, jak program ma reagować na konkretne rodzaje wyjątków. Główne metody w klasie `Exception` to:

1. `getMessage()`: Zwraca komunikat opisujący wyjątek.
2. `printStackTrace()`: Wypisuje ślad stosu (stack trace) wyjątku, co jest przydatne do debugowania.

Oprócz klasy `Exception`, w Javie istnieje wiele innych klas wyjątków, które dziedziczą z `Exception`, takie jak `RuntimeException`, `IOException`, `NullPointerException`, itp. Programista może także tworzyć własne klasy wyjątków, dziedzicząc je z `Exception` lub innych klas wyjątków, w zależności od potrzeb.

Jak działa obsługa wyjątków w Javie:

1. W kodzie źródłowym programu programista używa bloków `try-catch` do otoczenia kodu, który może wygenerować wyjątek. Blok `try` zawiera kod, który ma zostać wykonany, a blok `catch` zawiera kod, który zostanie wykonany w przypadku zgłoszenia wyjątku.

2. Jeśli w bloku `try` zostanie wygenerowany wyjątek, Java próbuje dopasować go do odpowiedniego bloku `catch` na podstawie typu wyjątku. Jeśli pasujący blok `catch` zostanie znaleziony, kod w tym bloku zostaje wykonany.

3. Po zakończeniu obsługi wyjątku, wykonanie programu kontynuuje się poza blokiem `try-catch`.

4. Można również używać bloków `finally`, które zawierają kod, który zostanie wykonany niezależnie od tego, czy wyjątek został zgłoszony, czy nie. To jest przydatne, aby upewnić się, że pewne zasoby zostaną zwolnione niezależnie od tego, co się wydarzy.

Obsługa wyjątków pozwala programowi na bardziej kontrolowane i bezpieczne reagowanie na nieoczekiwane sytuacje, co jest istotne w rozwijaniu stabilnych i niezawodnych aplikacji.

Zmienne w try-catch maja zasięg blokowy.

Try-catch-finally :

Wykona się zawsze nawet gdy zwracamy coś z metody.