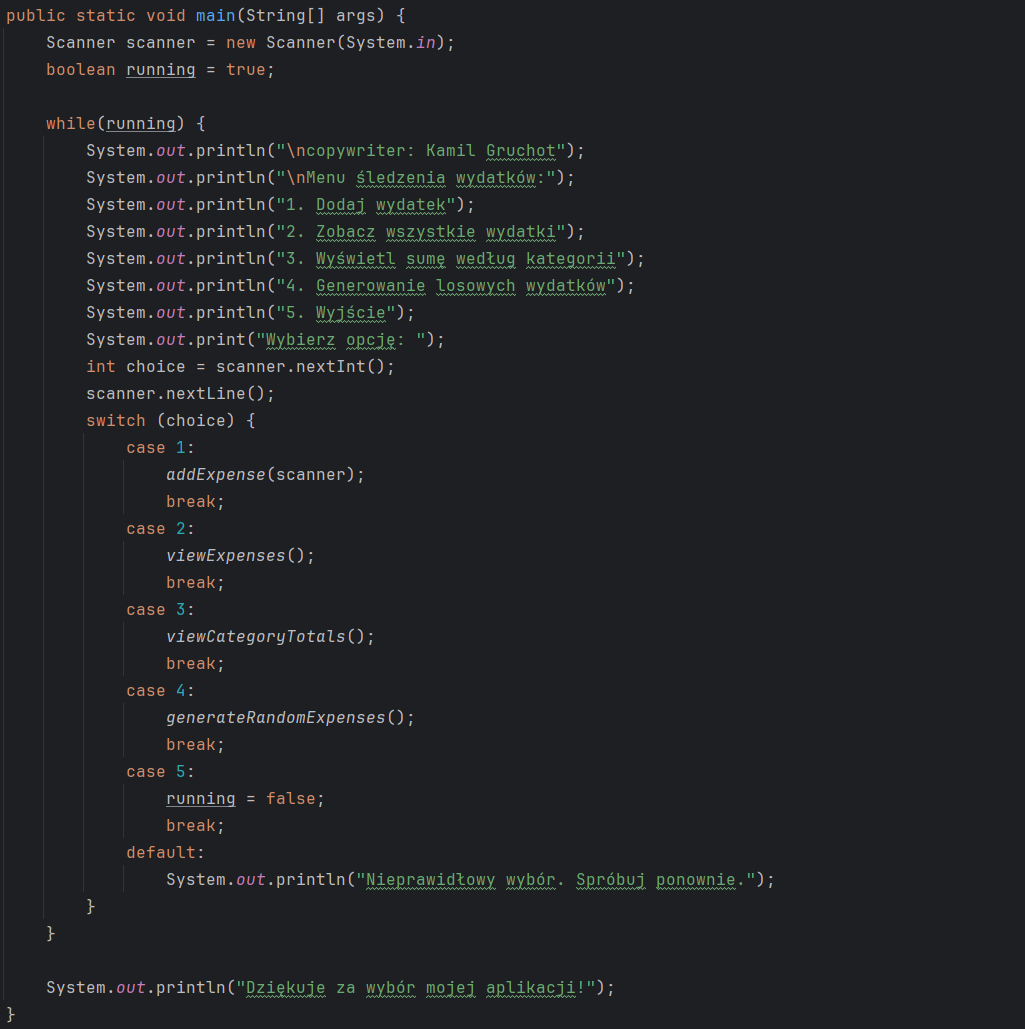
System śledzenia wydatów

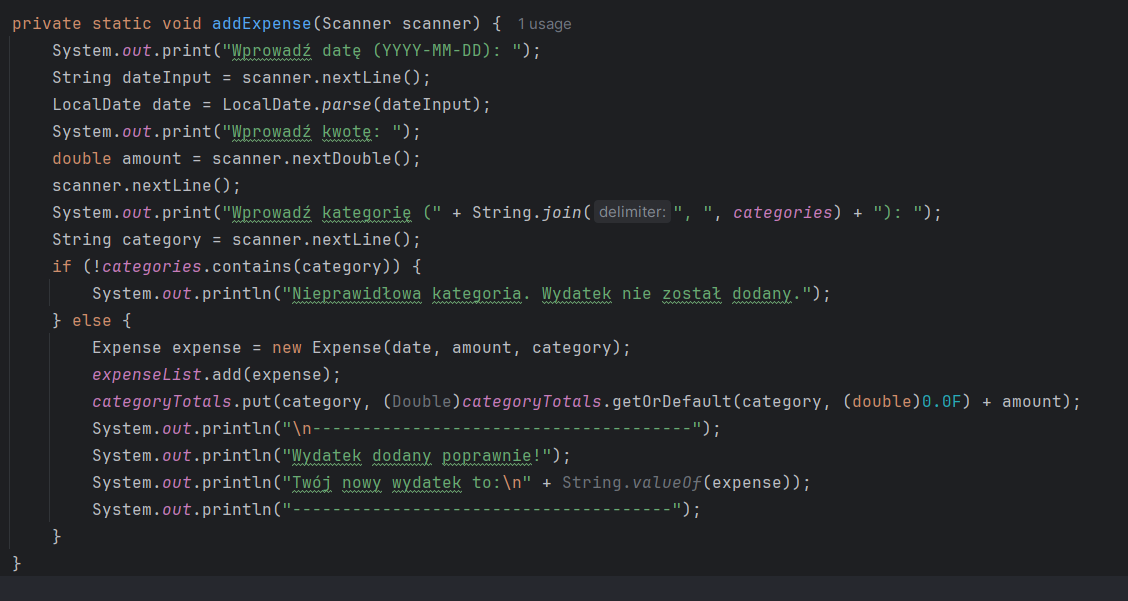


Lista “expenseList” przechowuje wszystkie wydatki dodane przez użytkownika w trakcie działania programu



Użycie obiektu “scanner”służy do odczytywania danych wprowadzonych orzez użytkownika w konsoli.Pozwala to na wprowadzenie zarówno liczb nextInt, jak i ciągów znaków nextLine.Natomiast zmienna running kontroluje działanie pętli jej wartość “true” sprawia że pętla **“while”** działa, a zmiana na “false” powododuje zakończenie programu.

“Switch” jako mechanizm wyboru analizuje wartość zmiennej “choice” i kieruje sterowanie do odpowiedniej części kodu w zależności od wybory użytkownika.



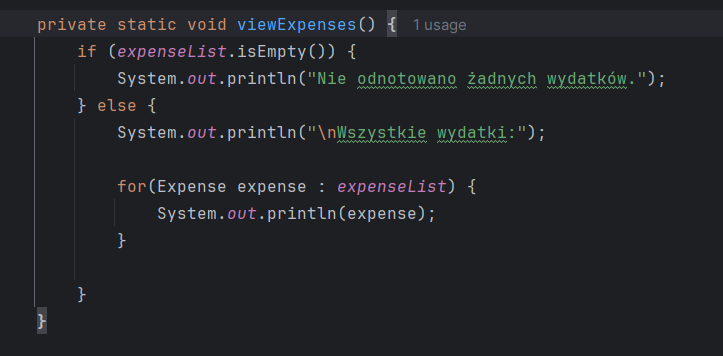
Odczytana wartość “string” jest konwertowana na obiekt typu “LocalDate” przy użyciu metody “LocalDate.parse” dzięki temu program może łatwo zarządzać datami i wykorzystywać je w różnych operacjach.

Program wyświetla dostępne kategorie wydatków pochodzące z zestawu “categories” i prosi o wybranie jednej z nich. Wprowadzana wartość jest zapisana jako ciąg znaków “string”.

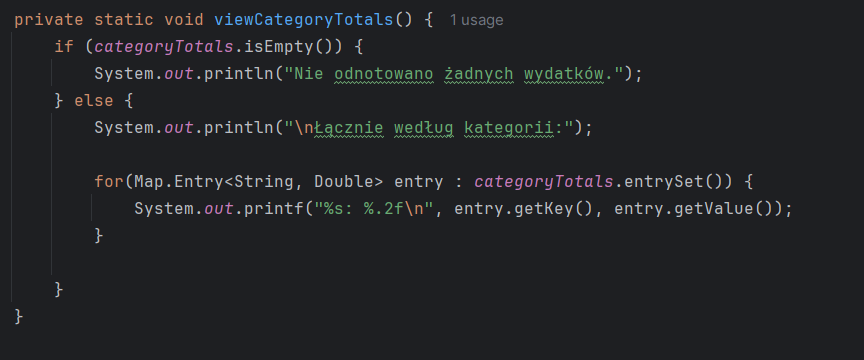
Dalej “if” sprawdza czy podana kategoria należy do zbioru “categories”.Jeśli kategoria jest nieprawidłowa,wyświetlany jest komunikat o błędzie a funkcja kończy działanie.

Mapa “categoryTotals” jest aktualizowana i zwiększana o wartość dodanego każdego wydatku.Jej wartość początkowa to 0.0.

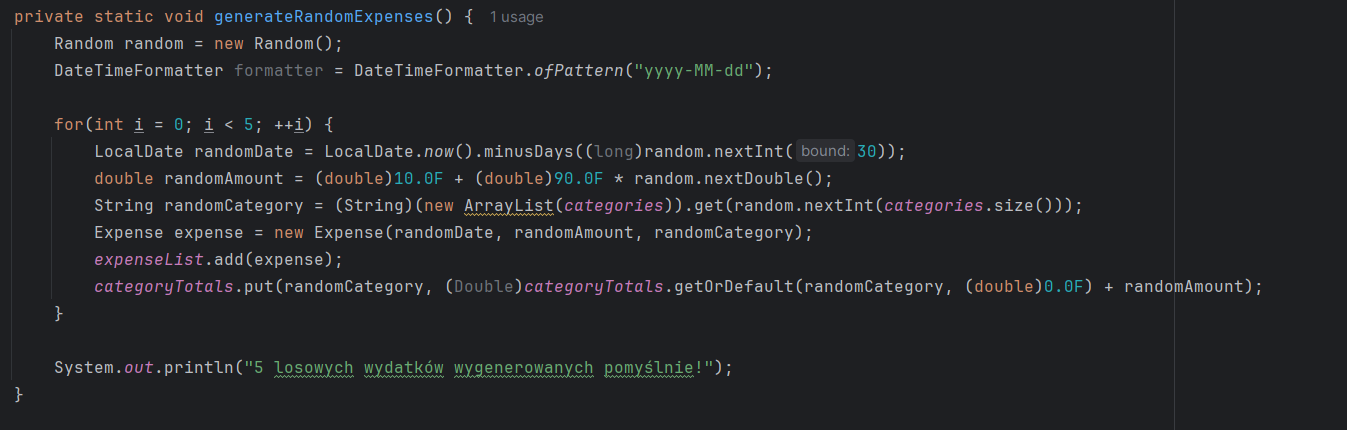
Po pomyślym dodaniu wydatku program wyświetla sczegóły nowego wydatku.Jako podsumowanie operacji.



Ten kod jest metodą “viewExpenses()”, która wyświetla listę wydatków. Jeśli lista wydatków “(expenseList)” jest pusta, wypisuje komunikat „Nie odnotowano żadnych wydatków”. Jeśli lista zawiera jakieś elementy, metoda wypisuje tekst „Wszystkie wydatki:” i następnie dla każdego obiektu “Expense” w “expenseList” wywołuje metodę “toString()” obiektu “Expense”, aby wyświetlić szczegóły wydatku.



Na początku sprawdzane jest, czy mapa categoryTotals jest pusta za pomocą metody isEmpty(). Jeśli mapa jest pusta, wypisywany jest komunikat „Nie odnotowano żadnych wydatków.”. Jeśli mapa zawiera dane, program wypisuje nagłówek „Łącznie według kategorii:” Następnie, za pomocą pętli for-each, iteruje po wszystkich wpisach w mapie (gdzie kluczem jest nazwa kategorii, a wartością suma wydatków w danej kategorii). Dla każdego wpisu wyświetla nazwę kategorii (entry.getKey()) oraz odpowiadającą jej sumę (entry.getValue()) w formacie liczby zmiennoprzecinkowej z dwoma miejscami po przecinku.



Random random = new Random(); - Tworzy obiekt klasy Random, który będzie wykorzystywany do generowania losowych liczb.DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd"); - Tworzy obiekt DateTimeFormatter, który ustala format daty (choć w tym kodzie formatowanie daty nie jest wykorzystywane bezpośrednio, ponieważ randomDate jest już typu LocalDate).

Pętla for wykonuje się 5 razy, aby wygenerować 5 losowych wydatków:

**Losowanie daty**: LocalDate randomDate = LocalDate.now().minusDays((long)random.nextInt(30)); — Losuje datę z ostatnich 30 dni (od dzisiejszej daty).

**Losowanie kwoty**: double randomAmount = (double)10.0F + (double)90.0F \* random.nextDouble(); — Losuje kwotę wydatku w przedziale od 10.0 do 100.0.

**Losowanie kategorii**: String randomCategory = (String)(new ArrayList(categories)).get(random.nextInt(categories.size())); — Losuje jedną kategorię z listy categories.

Po wygenerowaniu 5 losowych wydatków, na konsoli wypisywany jest komunikat: "5 losowych wydatków wygenerowanych pomyślnie!".



private LocalDate date; — Pole typu LocalDate, które przechowuje datę wydatku. Używa się go do zapisu daty w formacie yyyy-MM-dd.

private double amount; — Pole typu double, które przechowuje kwotę wydatku. Jest to liczba zmiennoprzecinkowa, która może zawierać dwie cyfry po przecinku (np. 25.50).

private String category; — Pole typu String, które przechowuje kategorię wydatku (np. „żywność”, „transport”).

Konstruktor klasy Expense przyjmuje trzy argumenty: datę (LocalDate), kwotę (double) i kategorię (String). Inicjalizuje on odpowiednie pola obiektu.

Używa ona formatu String.format(), aby stworzyć sformatowany tekst, w którym:

this.date reprezentuje datę wydatku.

this.amount jest kwotą wydatku, która jest sformatowana z dwoma miejscami po przecinku (np. 25.50).

this.category to kategoria wydatku.