# Zaawansowane programowanie obiektowe
# Lab. 7
# (Refleksja, adnotacje, pliki)

1 (REFLEKSJA, 1 pkt.)
Proszę napisać klasę o nazwie MaxSearchAlgorithms, która będzie zawierała 3 algorytmy (jako osobne metody o domyślnej/pakietowej widoczności) szukania maksimum w tablicy liczb typu int: od lewej do prawej, od prawej do lewej oraz najpierw czytane elementy na indeksach parzystych, a następnie na nieparzystych (uwaga: nie należy zakładać, że tablica przekazana jako argument zawiera parzystą bądź nieparzystą liczbę elementów). Wszystkie te 3 funkcje mają zwracać tablicę/listę (np. ArrayList) elementów, które były „chwilowymi" elementami maksymalnymi przy odnośnej strategii skanowania.

Przykład: jeśli na wejściu funkcji skanującej od prawej do lewej będzie tablica {4, 9, 3, 7, 4, 1, 5, 2}, to funkcja ma zwrócić tablicę {2, 5, 7, 9}.

Opisane 3 metody mają być wywołane na rzecz obiektu klasy MaxSearchAlgorithms utworzonego w innej klasie, za pomocą mechanizmu refleksji. Należy mianowicie odczytać wszystkie metody zdefiniowane w tej klasie, a następnie „odfiltrować" te, które nie zawierają w nazwie ciągu „Scan" (opisane wyżej 3 metody będą taki ciąg znaków w nazwie posiadały). Metody zawierające ww. napis mają zostać uruchomione (metoda invoke(...) z klasy Method), a zwrócone przez nie tablice wypisane na ekran.

2 (REFLEKSJA I ADNOTACJE, 1 pkt)
Utwórz klasę posiadającą przynajmniej 4 pola (prywatne) różnych typów (w tym prymitywnych), np. int, String, boolean...
Wygeneruj, przy użyciu Eclipse'a, settery i gettery dla tych pól.
Następnie napisz na dwa sposoby metodę equals(...) porównującą obiekty Twojej klasy, uwzględniając wszystkie pola z wyjątkiem jednego (np. przy klasie Osoba można zignorować pole telefon):
rozwiązanie tradycyjne („ręczne" porównywanie pól),
z użyciem refleksji i adnotacji.

Ad a) Opatrz metodę equals(...) odpowiednią adnotacją oznaczającą metodę przeciążoną (z Object).

Ad b) Dodaj własny typ adnotacyjny:

```
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.METHOD)
@interface IgnoreEquals {}
```

a następnie wykorzystaj go przy wybranym polu stosując odpowiednią metodę z klasy Method.

Rozwiązanie przetestuj.

3. (0.5 pkt)
Napisz program generujący *N* liczb losowych o rozkładzie normalnym (gaussowskim) o zadanych parametrach *m* (średnia) i $\sigma$ (odchylenie standardowe) i zapisujący je do pliku binarnego. Następnie plik ten jest odczytywany, liczba po liczbie, a dane zapisywane do nowego pliku (tekstowego), każda liczba w osobnym wierszu. Dodatkowo należy zwizualizować (np. przy pomocy Excela) histogram tych liczb, aby sprawdzić, czy rzeczywiście rozkład przypomina krzywą dzwonową Gaussa. Zapis do pliku tekstowego ma wykorzystywać konwencję formatu przyjętą w Polsce (tj. przecinki zamiast kropek dziesiętnych).
Parametry *N* (liczba całkowita), *m* i $\sigma$ (liczby zmiennoprzecinkowe) mają być argumentami wiersza poleceń. Przy pomocy mechanizmu asercji dopilnuj, aby odchylenie standardowe nie było liczbą ujemną.

Wskazówki.
a) Generacja liczb o rozkładzie normalnym – poszukać odpowiedniej metody w klasie Random.
b) Zapis/odczyt liczb w postaci binarnej: klasy DataOutputStream / DataInputStream.
c) Ustawienia lokalne Polski – można wykorzystać ustawienia Niemiec (pole statyczne Locale.GERMANY), a następnie utworzyć odpowiedni obiekt klasy NumberFormat.