Zadanie 1

Zaimplementuj kolejkę (ang. queue), w której mogą być przechowywane dowolne obiekty.
Implementacja kolejki powinna być zgodna z interfejsem Queue (Queue.java).
W celu sprawdzenia poprawności, wykorzystaj test QueueTest.java.
Użyj generyków.

Zadanie 2

Zaimplementuj stos (ang. stack), w którym mogą być przechowywane dowolne obiekty.
Implementacja stosu powinna być zgodna z interfejsem Stack (Stack.java).
W celu sprawdzenia poprawności, wykorzystaj test StackTest.java.
Użyj generyków.
Jeśli stos jest pusy, metody peek() i pop() powinny zwracać IllegalStateException.

Zadanie 3

Napisz dwie aplikacje konsolowe z wykorzystaniem JMS.

a) Pierwsza aplikacja – klient – powinna łączyć się do tematu (ang. topic) i wyświetlać wiadomości tekstowe wysyłane przez serwer (producent). W momencie otrzymania wiadomości „/exit", aplikacja klienta powinna przerwać działanie.
b) Druga aplikacja – serwer – powinna łączyć się do tematu (ang. topic) i umożliwić użytkownikowi wysyłanie wiadomości wpisanych w konsoli. W momencie wpisania przez użytkownika wiadomości „/exit", aplikacja powinna wysłać wiadomość do topicu i zakończyć działanie.

Wykorzystaj ActiveMQ (np. używając Dockera: $ docker run --name='activemq' -p 8161:8161 -p 61616:61616 -p 61613:61613 webcenter/activemq, konsola administracyjna: http://localhost:8161/admin/index.jsp, login admin, hasło admin).
Przetestuj aplikację uruchamiając jeden serwer i co najmniej dwóch klientów. Do każdego z klientów powinna dotrzeć wiadomość wysłana przez serwer.

Zadanie 4

Napisz aplikację webową, która po wpisaniu w formularz adresu (nazwa miasta, ulicy, kodu pocztowego – jedno pole) odpyta Google Maps API o współrzędne tego miejsca i na następnej stronie wyświetli mapę Google Maps z zaznaczonym punktem / obszarem.
Wykorzystaj https://developers.google.com/maps/documentation/geocoding/intro

Zadanie 5

Napisz aplikację webową, która po wpisaniu w pole formularza adresu URL zwróci użytkownikowi kod HTML podanej strony.

a) W przypadku, gdy podana strona nie istnieje lub adres nie jest poprawnym adresem WWW, aplikacja powinna wyświetlić zrozumiały opis problemu.
b) W przypadku, gdy podany adres WWW nie prowadzi do strony, która zawiera kod HTML, aplikacja powinna poinformować o tym użytkownika.
c) Jeśli adres jest poprawny i prowadzi do poprawnej zawartości, użytkownik powinien zobaczyć kod HTML, a nie wynik parsowania kodu HTML przez przeglądarkę.

Zadanie 6

Napisz aplikację konsolową, która pobierze od użytkownika adres WWW i sprawdzi dostępne metody http i rozmiar zasobu.

a) Użytkownik powinien mieć możliwość podania adresu jako parametr wywołania programu lub bezpośredni po uruchomieniu programu (Scanner)
b) Jeśli podany adres nie istnieje, aplikacja powinna poinformować o tym użytkownik i zakończyć działanie z kodem wyjścia innym niż 0.
c) Jeśli podany zasób jest dostępny, aplikacja powinna wyświetlić dostępne metody http oraz rozmiar zasobu.
d) Aplikacja nie powinna fizycznie pobierać zasobu wskazanego przez adres.

Wykorzystaj metody http: HEAD, OPTIONS

Zadanie 7

Napisz aplikację webową, która będzie symulować spłatę kredytu hipotecznego. Aplikacja powinna umożliwić podanie poniższych danych:
- kwotę kredytu
- okres kredytowania (w miesiącach)
- marżę banku (w procentach)
- prowizję banku (w procentach)

Po wpisaniu w/w danych, aplikacja powinna wyświetlić tabelę z symulacją spłaty kredytu:
- numer miesiąca
- kwotę raty
- kwotę pozostałą do spłaty

Nad tabelą powinna pojawić się informacja o RRSO i o całkowitym koszcie kredytu (kapitał + odsetki). Zakładamy, że raty są równe a oprocentowanie nie zmienia się w czasie.

Zadanie 8

Napisz aplikację konsolową, która umożliwi pomniejszenie podanego zdjęcia o połowę.

a) Plik źródłowy i plik docelowy powinny być podane jako parametry wywołania programu.
b) Wymiary pliku docelowego powinny być połową wartości wymiarów pliku źródłowego, tj. plik źródłowy o wymiarach 800x600 powinien zostać przeskalowany do pliku docelowego o wymiarach 400x300
c) Użyj klas Graphics / BufferedImage / ImageIcon lub dowolnej biblioteki do operacji na plikach graficznych.

Zadanie 9

Napisz aplikację konsolową, która skompresuje do formatu zip zawartość podanego folderu.

a) Ścieżka do folderu powinna zostać przekazana jako argument wywołania aplikacji.
b) Ścieżka do pliku wynikowego (zip) powinna zostać przekazana jako argument wywołania aplikacji (opcjonalnie). Jeśli nie zostanie podana, paczka powinna być utworzona na tym samym poziomie co kompresowany folder, z nazwą nazwaFolderu.zip
c) Do paczki zip powinny trafić tylko pliki znajdujące się bezpośrednio w podanym folderze. Nie uwzględniamy plików / folderów zagnieżdżonych głębiej.

Zadanie 10

Napisz metody do konwersji:
- temperatura w stopniach Celsjusza <-> w stopniach Fahrenheita
- odległości w kilometrach <-> w milach
- wagi w kilogramach <-> w funtach

Zaimplementuj interfejs UnitConverter.java. Aby przetestować swoje metody, wykorzystaj test UnitConverterTest.java.