Burger maker

Twoim zadaniem jest napisanie symulatora półautomatycznego kucharza burgerów.

Kucharz otrzymuje po jednym (losowo wygenerowanym) zamówieniu na raz.

W przypadku gdy kucharzowi uda się w **odpowiedniej kolejności** ułożyć **zadane składniki** dostaje punkt, w przeciwnym razie traci punkt. Pamiętaj o kolejności w jakiej składniki lądują na kanapce (są nakładane na górę).

Na realizację wszystkich zamówień kucharz ma **60 sekund**, po tym czasie następuje **zatrzymanie timera** oraz **wyświetlenie wyniku.**

Kucharz ma do dyspozycji przyciski do dodawania poszczególnych składników do kanapek, a także przycisk „**Opróżnij talerz**" opróżniający talerz w razie nałożenia błędnych składników oraz „**Gotowe**" w celu wysłania zamówienia „na salę" (wtedy następuje walidacja czy kuchcik użył odpowiednich składników).

Zamówienia składają się od **3** do **6** składników. Klient nasz Pan, dlatego pozwalamy zamawiać kombinacje typu mięso, pomidor, mięso (nie przejmuj się tym w tej chwili w trakcie pisania generatora zamówień.

1. Na podstawie udostępnionego szablonu stwórz aplikację okienkową opisaną powyżej
2. Dodaj czytelne nazwy składników (zamiast np. wyświetlać „CUCUMBER" wyświetlaj ogórek
3. Składniki muszą znajdować się w bułce (dowolnego typu), nie obsługujemy już kombinacji typu mięso, pomidor, mięso. Być może część wartości enumów trzeba przenieść do innej klasy? albo dodać nowe pole do enuma?
4. Sanepid odkrył, że podajemy hamburgery z surowym mięsem, od tej chwili musimy smażyć burgery przed podaniem. Po naciśnięciu przycisku „mięso" przycisk powinien stawać się nieaktywny na czas smażenia i wyświetlać czas pozostały do usmażenia mięsa, potem znowu staje się aktywny, a mięso dodaje się na listę składników.
5. Od tej chwili każdy kawałek mięsa w kanapce to +2 punkty. Może rozsądnie będzie dodać pole do enuma?
6. *Dodaj obrazki na przyciskach obrazujące co dany przycisk dodaje.

Wskazówki:

Wyświetlanie dialogu:

```
Alert alert = new Alert(Alert.AlertType.INFORMATION);
alert.setTitle("Koniec gry");
alert.setHeaderText("Gratulacje!");
alert.setContentText(SCORE_TOKEN + gameScore);
alert.showAndWait();
```

Odliczanie czasu można zrobić na przykład tak:

```
scheduledExecutorService = Executors.newSingleThreadScheduledExecutor();
timer = scheduledExecutorService.scheduleAtFixedRate(() -> decrementGameTime(), 1, 1,
TimeUnit.SECONDS);
```

Uwaga, wszystkie modyfikacje komponentów (np. label), muszą być przeprowadzane w głównym wątku, żeby to zapewnić użyj w metodzie wywoływanej przez scheduler:

```
Platform.runLater(() -> to co chcę zrobić z UI)
```

Na przykład:

```
Platform.runLater(() -> label.setText(...))
```

Wyświetlanie dialogu:

```
Alert alert = new Alert(Alert.AlertType.INFORMATION);
alert.setTitle("Koniec gry");
alert.setHeaderText("Gratulacje!");
alert.setContentText("Jakas wiadomosc!");
alert.show();
```

Instrukcja:

1. Rozejrzyj się po projekcie. Pamiętaj, że w razie gdybyś wprowadziła/wprowadził jakiekolwiek psujące zmiany możesz cofnąć je przy pomocy gita
2. Klasa Order powinna zawierać listę składników, które użytkownik zamawia (na tą chwilę tylko jedno pole)
3. Klasa CurrentPlateHandler odpowiada za obsługę aktualnego zamówienia (dodawanie i usuwanie składników):
    a. Metoda addIngredient powinna pozwalać na dodanie składnika do aktualnie komponowanej kanapki
    b. Metoda clearPlate powinna usuwać aktualnie budowaną kanapkę (w razie gdybyśmy się pomylili)
    c. Metoda validate wywołuje validate z klasy OrderValidator, to nic innego jak sprawdzenie czy kanapka, którą utworzyliśmy jest zgodna z oczekiwaniami klienta
    d. Wszystkie powyższe metody powinny operować na polu ingredientsOnPlate (to Stack) lub jeśli uważasz, że można to zrobić inaczej śmiało 😊
    e. W kodzie tej klasy umyślnie pozostawiłem błąd (znajdź go, jeśli go nie widzisz uruchom aplikację po implementacji metod w tej klasie i zobacz co się stanie)
4. Klasa OrderValidator ma jedną metodę validate, która powinna sprawdzać czy podane składniki spełniają życzenie klienta i **są ułożone w odpowiedniej kolejności!**
5. W klasie OrderGenerator utwórz metodę generate(), która będzie generowała losowo nowe zamówienie, ilość składników w nim to co najmniej **3**, ale nie więcej niż **6**. Składniki mogą się powtarzać i nie muszą być zamknięte w bułce.
6. Kliknięcia poszczególnych przycisków z nazwą składniku powinny dodawać ten właśnie składnik (np. przycisk „ogórek” powinien dodawać ogórka do currentPlateHandler za pomocą metody addIngredient) oraz wyświetlać w labelce onPlate (za pomocą onPlate.setText("jakiś tekst\na ten jest w nowej linii")) zaktualizowaną listę składników. Listę składników możesz wziąć z currentPlateHandler.getIngredients()

7. Naciśnięcie „Wyczysc talerz” powinno czyścić talerz, a więc wywoływać currentPlateHandler.clearPlate()
8. Naciśnięcie przycisku „Gotowe” powinno zglaszac posiłek jako gotowy do wydania. Wtedy powinna się uruchomić walidacja (currentPlateHandler.validate(argumenty)). Jeśli currentPlateHandler.validate zwróci true dodajemy punkt, jeśli false odejmujemy. Pamiętaj o wyświetleniu zaktualizowanego wyniku (score.setText("Wynik:" + costam)) oraz o wylosowaniu kolejnego zamówienia.
9. Za odliczanie czasu zabierz się na końcu, zrobimy je wspólnie. Po upłynięciu 60 sekund powinno pokazać się okno dialogowe z informacją o wyniku (patrz sekcja wskazówki, żeby dowiedzieć się jak pokazać takie okno).