HO
GENT

Collections - Werkcollege

Table of Contents

1. HashSet Maaltafels

Maak twee HashSets met integers, de ene vul je via een for loop op met de maaltafel van 2 (2 - 20) en de andere vul je op dezelfde manier met de maaltafel van 3 (3 - 30). Print beide sets uit in de console.

Kopieer de waarden van de tafel van twee nu in een nieuwe HashSet en voeg er daarna de de tafel van drie HashSet aan toe met de `UnionWith()` methode. Print het resultaat van de combinatie uit in de console.

Wat valt er op? Als je niet direct iets opvalt kan je dezelfde oefening eens maken met een List<> en de outputs vergelijken.

2. Elementen Dictionary

Maak een klasse Element aan, de klasse bevat drie properties:

1. Symbol: string
2. Name: string
3. AtomicNumber: int

Maak een Dictionary aan met string als key en element als value. Vul daarna de dictionary met deze vier elementen waarbij je de symbol waarde als key gebruikt en het volledig element object als value.

1. H (symbol), Waterstof (name), 1 (atomic number)
2. He, Helium, 2
3. Li, Lithium, 3
4. Be, Berilium, 4

Vraag aan de gebruiker om een symbol in te geven via de command line, ga daarna op zoek in de dictionary of er een symbol bestaat dat overeen komt met de input van de gebruiker

- Indien ja print je alle properties van het element uit.
- Indien niet toon je de gebruiker een foutmelding.

Er zijn minstens drie verschillende manieren om het opvragen van een onbestaande key in een dictionary op te vangen, probeer er drie te vinden.

3. View Navigation

```
Currently on view: Home
Do you want to 'navigate', go 'back' or print 'history'?
```

Maak een enum met vier views (Splash, Home, Settings, Detail). Maak ook een queue of stack aan (beslis na het lezen van de opgave welke van toepassing is) die we history noemen. Voeg hier al twee waarden aan toe: Splash & Home om de start van onze applicatie te simuleren.

Onze applicatie zal kunnen navigeren tussen deze vier views. Zoals je in bovenstaande screenshot ziet zal onze applicatie drie verschillende functies hebben die je uitvoert als de gebruiker het overeenkomstige woord intypt:

1. **Navigate**: ga naar een view. Vraag aan de gebruiker naar welke van de vier views uit de enum hij/zij wil gaan. Navigeren naar de splash view mag niet, toon een fout wanneer de gebruiker dit probeert of foutieve input geeft en vraag opnieuw om een andere view in te geven. Voeg de genavigeerde view toe aan de history.
2. **Back**: keer terug naar de vorige view. Gebruik de history om terug te keren naar de vorige pagina. Op deze methode moet navigeren naar de splash view ook onmogelijk zijn.
3. **History**: print uit welke items in de history zitten.

Toon een foutmelding indien de input van de gebruiker foutief is. De gebruiker krijgt na één van deze drie acties uit te voeren opnieuw de vraag welke van de drie acties hij/zij wil uitvoeren.

4. Printer

In deze oefening maken we een printer applicatie. Begin met een klasse printer te maken, deze heeft twee publieke functies:

1. `Print(string message)` voeg een tekst toe aan de lijst met te printen teksten. Indien de printer aan staat wordt het bericht direct geprint.
2. `PressPowerButton()` als de printer uit staat wordt hij aan gezet en vice versa. Als de printer aan wordt gezet moet hij direct alle teksten die in de lijst met te printen teksten staan uitprinten.

De interne werking van de klasse printer kan je zelf bepalen. Maak in je program klasse een instantie van het printer object aan. Je kan de twee bovenstaande functies aanroepen via user input in de command line. De tekst die de print functie nodig heeft kan de gebruiker zelf invoeren.

Beslis zelf welk type collection het beste is om de uit te printen teksten in op te slaan.