Færdighedsprøve

I Fagene

Software Construction

og

Software Design

1. semester juni 2022

Domæne beskrivelse

FDF afholder hvert 5 år landslejr for børn mellem 10 og 15 år ved Julsø i Jylland. I den periode hvor lejren afholdes er det ofte meget varmt og derfor vil lejrledelsen gerne give de enkelte kredse mulighed for at bade i den nærliggende sø. Lejrledelsen vil derfor gerne have et system kaldet **Badetidssystemet,** der kan administrere, hvornår de enkelte kredse må bade i søen.

I systemet skal der være en klasse **BadetidsPeriode.** Klassen skal indeholde følgende properties:

Property	**Datatype**	**Beskrivelse**
Type	string	Angiver typen af badeaktivitet – f.eks. ”Morgendukkert” eller ”Aftensvaler”. Default er ”Badning”
UgeDag	DayOfWeek	Dag i ugen hvor man må bade. Default er `DayOfWeek.Sunday`
StartTidspunkt	DateTime	Default er `new DateTime()`
SlutTidspunkt	DateTime	Default er `new DateTime()`

Ligeledes skal der oprettes en klasse **Kreds**, der indeholder oplysninger om de enkelte tilmeldte kredse.

Klassen ***Kreds*** har følgende properties:

Property	**Datatype**	**Beskrivelse**
ID	string	Kredsens ID
Navn	string	Kredsens navn
Adresse	string	Kredsens adresse
AntalDeltagere	int	Antallet af personer kredsen deltager med

Klassen **BadetidsPeriode** skal indeholde en dictionary af **Kreds**e **`, Dictionary<string, Kreds>,`** hvor ID er er key

Opgave 1. Opret Klassen Kreds

- Opret et nyt .NET Core console Application projekt, **Badetidssystemet.**
- Implementer klassen **Kreds** med instansfelter, properties og passende konstruktører.
- Implementer **`ToString()`** metoden på klasse.
- Test klassen ved at oprette nogle instanser af den i **`Main`** metoden og skriv dem ud til konsollen med **`Console.WriteLine`**

Opgave 2. Opret klassen BadetidsPeriode

BadetidsPeriode klassen skal som nævnt indeholde en dictionary af **Kreds** objekter, som skal instantieres i konstruktøren.

- Implementer klassen **`BadetidsPeriode`** med instansfelter, properties og passende konstruktører.
- Implementer **`ToString()`** metoden på klassen.
- Test klassen ved at oprette nogle instanser af den i Main metoder og skriv dem ud til konsollen med **`Console.WriteLine.`**

Hint:

- Et tidspunkt – f.eks. 06:30 kan i C# oprettes som
  `new DateTime().AddHours(6).AddMinutes(30)`
- Tidspunkt på dagen kan hentes med property **`TimeOfDay`** på klassen **`DateTime`**

Opgave 3. Lav klassediagrammet som det ser ud på nuværende tidspunkt

(hint – det skal være muligt at oprette, slette og ændre i aktiviteter og kataloget)

Opgave 4: Udvid BadetidsPeriode klassen med nogle CRUD metoder

- Tilføj metoden **`void AdderKreds(Kreds kreds)`** til klassen **BadetidsPeriode**.
- Tilføj metoden **`void SletKreds(string id)`** til klassen **BadetidsPeriode.** Metoden sletter **Kreds**en med det pågældende Id (= parameteren id).

- Test klassen **BadetidsPeriode** ved at oprette 2 instanser af den i Main metoden. Skriv hver enkelt instans ud til konsollen. Husk at teste begge metoder - både **`AdderKreds`** og **`SletKreds`** metoderne

Opgave 5: Lav et sekvensdiagram

Tegn et sekvensdiagram for et kald af metoden **`AdderKreds`**. Udgangspunktet for sekvensdiagrammet er **`Main`** metoden og sekvensdiagrammet skal kun vise et enkelt kald af **`AdderKreds`** og vise, hvorledes der oprettes et objekt af typen **Kreds** og at det tilføjes til dictionary i **Badetidssystemet**.

Opgave 6. Validering med betingende sætninger

Programmet skal sikre at følgende valideringsregler er overholdt:

1. StartTidsPunkt må ikke være senere end SlutTidsPunkt
2. Type skal være mindst 4 tegn lang
3. AntalDeltagere skal være større en 0

- Indfør betingede sætninger i klassernes metoder, der skriver en fejlbesked til konsollen, hvis en eller flere af ovenstående valideringsregler ikke er overholdt.

Hint:

- Valideringen kan ofte med udføres i **`set`**-delen af den pågældende property.

Opgave 7. Validering med betingende sætninger og exceptions

Ligesom ovenstående opgave, men nu skal programmet ændres på følgende måde:

1. I stedet for at skrive en fejlmeddelelse ud, hvis valideringsreglerne ikke er overholdt, skal metoderne smide (**`throw`**) en **`ArgumentException()`**
2. Test ændringerne med **`try catch`** i **`Main`** metoden.

Opgave 8. Lav dine egne user stories

Lav et par user stories, der relaterer sig til **Badetidssystemet.**

Du skal samtidig redegøre for INVEST kriterierne.

Opgave 9 Implementér dine egne user stories

Implementer de user stories, du definerede i opgaven ovenover.

Opgave 10 Nedarvning, for-loop og anvendelse af lister

Implementer en klasse **BadetidsPeriodeForLoopAndList,** der arver fra klassen **BadetidsPeriode.** Dette kan gøres vha. følgende forløb:

1. Opret klassen **BadetidsPeriodeForLoopAndList** med den krævede nedarvning.
2. Lav en konstruktør til **BadetidsPeriodeForLoopAndList.**
3. Lav metoderne **`AdderKreds`** og **`SletKreds`** virtuelle (erkær moderne med **`virtual`**) i klassen **BadetidsPeriode.**
4. Tilføj en liste af **Kreds**e - f.eks. **`List<Kreds> _kredseList`** - til klassen **BadetidsPeriodeForLoopAndList** og instantier den i konstruktøren.
5. Lav nedarvede udgaver af metoderne **`AdderKreds`** og **`SletKreds`,** der opererer på listen af **Kreds**e i klassen **BadetidsPeriodeForLoopAndList.** Husk at definere disse med **`override`.**
6. Implementer en **`ToString()`** metoden på klassen **BadetidsPeriodeForLoopAndList**.

Husk: Alle løkker i **BadetidsPeriodeForLoopAndList** skal implementers som **`for`**-loops.

Test **BadetidsPeriodeForLoopAndList** klassen i **`Main`** metoden, idet den skal fungere på samme måde som **BadetidsPeriode** klassen.