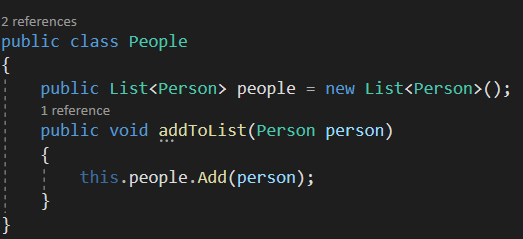
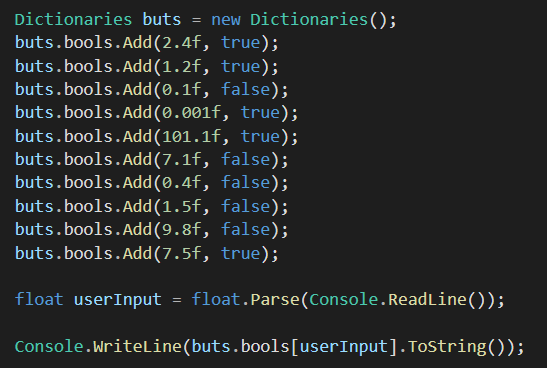
Vi har lært at lave Libraries hvilket, af hvad jeg kan forstå, er ting som tilføjer features til dig program, når du skal programmere. Det kan fx være alting fra en simpel function som fx en methode der parse en string, til en hel class der indeholder hundrede af values og har en masse methoder i den class.  
Dette kan blive brugt til at gøre ting nemmere, når du skal programmere. Hvorfor programmere noget indviklet til dit program, hvis der er en der allerede har gjort det for dig?



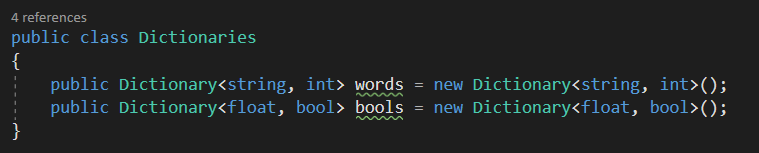
Her kan vi se, at jeg fx har lavet en class som hedder People. Og i denne class indeholder den en liste af people. Classen indeholder også en function som hedder addToList hvilket du kan bruge til at tilføje en person til listen. Dvs. at mit library indeholder en class der hedder People, men en indbygget function til at tilføje flere til dens liste.

Min mening: Jeg synes det var svært til at starte med, men synes at der er vildt fedt at have lært det, for det giver én en rigtig god ide til, hvordan tingene fungere, og hjælper én til at forstå hvordan programmering virkelig hænger sammen.

Vi har lavet Dictionaries, hvilket er noget man kan bruge til at gøre det lettere at finde ting. De indeholder ofte en key og en item. Key er det, som du skriver. Hvor item, er det som den låse op.



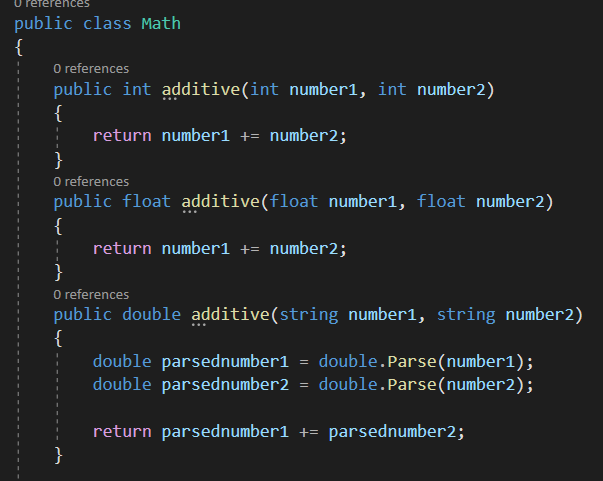
I dette eksempel kan vi se, at jeg først laver en dictionary, hvilket er en dictionaries classen, som indeholder dette:



I dette eksempel bruger jeg dictionariet bools, hvilket består af en float key, og bool item.

I det første billede kan vi se, at jeg laver en dictionary af typen bools, som hedder ”buts”. Så kan vi se, at jeg tilføjer en masse med forskellige tal, nogen som har en item der er true, og ngoen som er false.  
Efter det så parser vi det som brugeren skriver, og så skriver vi det, som passer til den key de har skrevet. Fx hvis de skrev 7.5, så ville der blive skrevet i console: True.

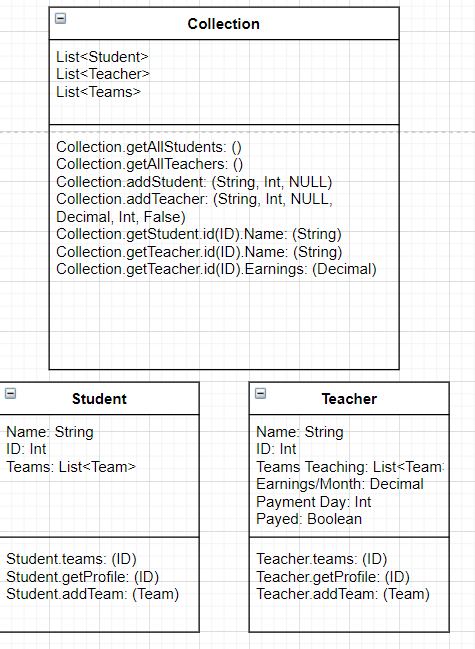
Min mening: Jeg kan se hvornår man ville bruge dette, men indtil videre har jeg ikke haft et projekt hvor dette har været nødvendigt.

Vi har arbejdet med noget som kaldes Overloading, hvilket er når du laver 2 methoder med samme navn, men modtager forskellige values. 

Som eksempel, så kan vi ser her at jeg har lavet 3 functioner med navnet ”additive”. Men de alle tager imod forskellige values. Dvs. hvis jeg skrev strings i parametrerne, så ville den vide at det er den 3. additive den skal køre, hvor den returnerer en double.

Min mening: Jeg synes det var super sjovt at bruge, og brugte det også i mit projekt i slutningen af ugen. Det kan blive lidt uoverskueligt hvis man ikke håndterer det rigtigt, så det skal man være forberedt på.

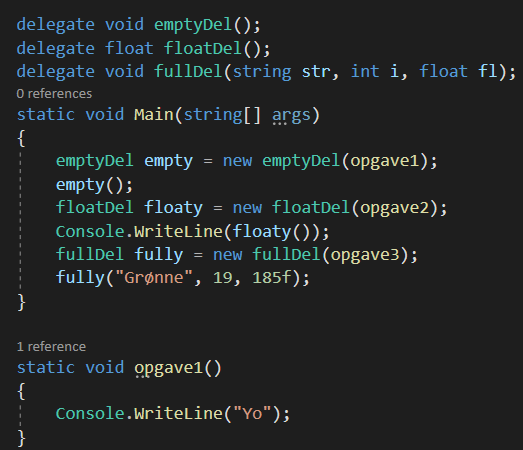
Vi lavede også UML diagrammer, hvilket er rigtig gode til at få overblik over projektek, før du starter. De indeholder class name, værdier og functioner.



For eksempel her kan vi se en class med navn ”Collection”, hvor den indeholder værdier som fx en liste af students.  
Under det kan vi se de methoder classen indeholder, som fx getAllStudents.

Min mening: Jeg synes at UML diagrammer tager semi lang tid at lave, me synes også at det hjælpe én pisse meget med planlægning og når ma starte med at bygge på sit program.

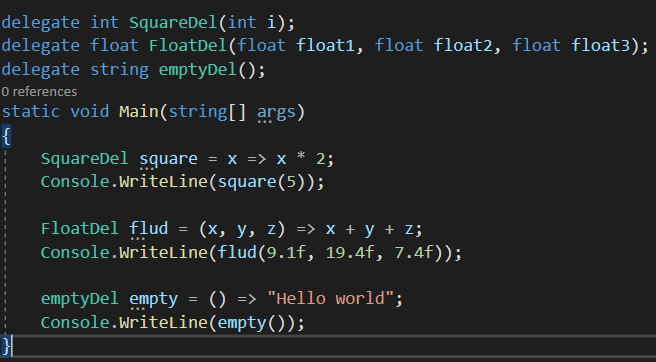
Vi har også arbejdet med Delegates, hvilket er en type værdi, som indeholder functioner og er klar til at køre dem. Du kan altid tilføje og fjerne functioner fra dem, hvilket kan hjælpe hvis du vil gøre det næmmere at køre en sequence af events.



For eskempel så kan vi se her at jeg laver en delegate class/function som hedder emptyDel. Jeg har derfter lavet en function som hedder opgave1, hvilket ikke tager imod nogen parametre og som returner en void. Som vi kan se, så tager emtpyDel heller ikke mod nogen parametre, og returner også en void. Dvs. at opgave1 kan bruges af emptyDel.  
Derefter kan vi se at jeg laver en ny emptyDel værdi som hedder empty, og indeholder opgave1. Derefter kører jeg den, og i consollen ville vi så kunne se, at den skriver ”Yo”.

Min mening: Jeg synes at deletages at geniale og har helt sikkert tænkt mig at bruge dem næste gang jeg laver et spil.

Lambda Expressions er en anden måde at lave methoder på.



He kan vi fx se at jeg laver en delegate som returner en int der hedder SquareDel. Inde i main kan vi se at vi laver en function af Square del classen som vi kalder square, hvor vi siger at den tager imod parametreren x, og i functionen siger den x \* 2. Derefter kører vi den og indtaster 5. Så skriver den i consollen: ”10”.