## Forarbejde

### Forarbejde til Rutediagrammet

Lavede et rutediagram for planetsimulationen. Først: find de vigtigste trin (setup, input-check, simulering, validering, afslutning). Så: beslutningspunkter for fejlrettelser. Justeringer = bedre flow. Resultat? Et skudsikkert overblik, perfekt til debugging og udvikling. ChatGPT er blevet brugt til valg af medie og overblik, over rækkefølge.

### Rutediagrammet

Vores rutediagram viser hvordan et program, vores program ville simulere planeternes bevægelse baseret på de forskellige parametre som position, hastighed og masse. Processen vil starter med initialiseringen af de forskellige parametre, efterfulgt af en kontrol af inputdataen. Vi har lavet det så hvis der bliver fundet/opstår fejl, rettes de før simuleringen fortsættes.

Når inputdataen er godkendt, vil programmet oprette simulationsspecifikationerne, som beskriver de nødvendige beregninger og betingelser for simuleringen. Derefter starter selve simuleringen, hvor planeternes positioner og hastigheder løbende opdateres. For at give brugeren et bedre overblik har vi inkluderet en visuel fremstilling af planeternes baner, som gør det nemmere at følge simulationens forløb.

Under simuleringen vurderes det løbende, om resultaterne er valide, og om der er behov for ændringer. Hvis der opstår fejl eller utilfredsstillende resultater, vil programmet automatisk køre flere iterationer for at forbedre simuleringen. Hvis resultaterne fortsat ikke er korrekte, stopper programmet simuleringen og genererer en fejlrapport.

Når simuleringen enten er godkendt eller afsluttet på anden vis, udarbejder programmet en slutrapport, som dokumenterer simulationens forløb og resultater. Vi har også tilføjet en funktion, der kan udføre afsluttende beregninger, hvis det er nødvendigt.

Samlet set har vi designet vores program med fokus på en struktureret arbejdsgang, hvor vi har lagt vægt på fejlretning, datavalidering og tydelig visualisering for at sikre en effektiv og brugervenlig simulering af planeternes bevægelse.

### Forarbejde til pseudokode

Før vi kunne gå i gang med at skrive pseudokoden skulle vi først forstå de centrale principper bag vores program. Det er ting som at forholdet imellem planeten og stjernen skal følge newtons gravitationslove, hvilket vi kan bruge til at opdatere planetens bane. Herefter har vi defineret hvilke komponenter vores program skal bestå af. Vi har to legemer, altså stjerne og planet, vi har tyngdekraften der skal defineres og så har vi nogle parametre i funktionerne der skal kunne opdateres som er massen af begge legemer og tyngdekraften imellem dem. Til inspiration af pseudokodens opstilling er der blevet brugt ChatGPT, men selve ideerne og arbejdet bag er selv lavet.

### Pseudokoden

Pseudokoden er skrevet som en slags opskrift til programmet. I starten skal konstanten i Newtons formel defineres. Herefter laves der en “setup” hvor brugeren skal indsætte værdierne til parametrene nævnt tidligere og baggrundsstørrelsen. “Draw” bruges til at tegne solen ind som midtpunktet og til at tegne ellipsebanen som simulerer planetens bevægelse. Her er det blevet valgt at ellipsen skal laves i et 1/0.6 forhold da det så ligner de fleste planeters bane. Her er radius som der bliver regnet ud fra, den store halvakse. Farverne indsat i pseudokoden er gul til stjernen, hvid til ellipsen og sort til baggrunden.

Herefter kommer funktionerne, hvor vi har en i vores program.  den bruges til at udregne ellipsens radius som bruges i “draw”. For at sørge for at radius af ellipsen er i forhold til newtons lov har vi isoleret for radius i formlen. Dette giver som sagt den halve storakse, som er den længste radius i ellipsen. Den anden radius har vi gjort 0.6 gange så stor.