Teorioppgave 1 – Exception:

Exceptions betyr unntak, men oppfører seg mer som forventinger. Exceptions brukes ofte for å gi tilbakemeldinger på forventede feilmeldinger. En typisk exception kan være «ValueError». Koden som kan utløse en exception, plasseres i try-blokken, og koden som skal håndtere exceptionen plasseres i except-blokken. Man kan ha flere except-blokker for å håndtere ulike typer exceptions på forskjellige måter. Dette lar deg ta spesifikke tiltak basert på hvilken type exception som oppstår. Under er et eksempel på bruken av «ValueError» hvor den slår ut hvis brukeren ikke skriver inn et heltall: Et bilde som inneholder tekst, skjermbilde, display, Font

Automatisk generert beskrivelse

Noen grunner til å bruke except, er feilsøking, for å sikre at programmet ikke krasjer når det oppstår feil, og for feilsøking og retting av feil i koden.

Teorioppgave 2 – Klasse:

En klasse en blokk med kode som brukes for å opprette objekter. Objekter brukes for å kalle på en funksjon/metode i klassen. En klasse skal alltid ha et navn som starter med stor forbokstav. Dette gjelder alle kodespråk, og er noe som gjør klassene lett gjenkjennelige. Videre har en klasse en konstruktør (\_\_init\_\_) som oppretter en ny instans av klassen. Den tar inn parametere som skal brukes i klassen. I dette tilfellet var det «lengde» og «bredde». «Self» skal alltid være med i klasser, ettersom det er den som brukes til å definere attributtene. Videre i klassen finner man metoder / funksjoner. Disse metodene tar ingen ekstra parametere, og brukes kun til å utføre oppgaver som er spesifikke for objekter av gjeldene klasse. Utenfor klassen kan man også lage variabler eller instanser av klassen, slik at det blir mer oversiktlig. Til slutt må man kalle på både klassen/variabelen og funksjonen man vil bruke innenfor klassen. Utsnittet under viser en fullstendig klasse hvor det også kalles på en metode fra klassen.

Et bilde som inneholder tekst, programvare, Multimedieprogramvare, skjermbilde

Automatisk generert beskrivelse

Teorioppgave 3 – Objekt:

I forrige oppgave ble det nevnt instans og variabel. En bedre terminologi på det er faktisk et «objekt». Disse objektene ligger ikke inne i klassen, men er heller noe man oppretter utenfor og styrer utfallet av hva som blir returnert i en eventuell «print». Hvis vi igjen ser på kodeeksempelet fra forrige oppgave, ser vi at linje 9 inneholder et objekt. Objektet ble navngitt «areal\_instans», men kunne blitt navngitt noe helt annet. Det har egentlig ingenting å si. Å opprette et objekt og gi det et rimelig navn gjør det lettere og mer oversiktlig å bruke det senere eller flere ganger. Vi ser også objektet definerer de to pålagte parameterne som i dette tilfellet ble 5 og 5. Man kan lage så mange objekter man vil, da helst med forskjellige verdier og navn, ellers risikerer man å overskrive det man allerede har definert.

Programmeringsoppgave 2 - Dokumentasjon

**Oppgavevalg**

Da vi fikk utdelt oppgaven og fikk opplyst at det var to fremgangsmåter var jeg først innstilt på at jeg kun skulle gå for «fremgangsmåte 1». Tingen var at jeg aldri hadde spilt Blackjack, og viste egentlig at poenget med spillet var å få 21. Jeg så meg derfor nødt til å starte med «fremgangsmåte 2».

**Oppstart:**

Jeg lastet modulen som innehold alle kortene, kortverdiene og de ulike funksjonene. Jeg leste gjennom hele oppgaven og måtte til slutt spørre Chat GPT om en kjapp introduksjon til hva blackjack var og hvordan det skulle spilles. Etter å ha forstått meg litt mer på blackjack begynte jeg å lage et deck med tilfeldige kort, hvor jeg egentlig bare lånte funksjonen fra modulen. Deretter Lagde jeg to tomme lister, én for spilleren (meg) og en for dealeren. Jeg lagde deretter en funksjon som skulle dele ut kort til disse listene, og dermed spilleren og dealeren. Jeg tok også to parametere i funksjonen, «player\_amount» og «dealer\_amount». Denne skulle brukes hver gang spillet starter hvor begge parter får to kort, men også i hit eller stand-situasjonene, hvor enten dealeren eller spilleren skal få ett kort av gangen.

Etter å ha delt ut kort til begge partene skulle spilleren få vite kortene sine og håndverdien sin. For ryddighetens skyld lagde jeg derfor en funksjon som skrev ut alle kortene med en liste som parameter. Brukeren skulle også få vite det første kortet til dealeren og verdien til dette kortet. Her hentet jeg bare ut det første kortet og verdien ved hjelp av indeksering.

**Hoveddel:**