Reeksamen i Programmering og Problemløsning

Eksamen vil blive afhold tirsdag d. 19/4 og onsdag d. 20/4, og eksamensformen er

  30 minutters mundtlig eksamen (inklusive bedømmelse, uden forberedelse) i fuldt pensum. Skriftlige hjælpemidler (notater og bøger mv.) er tilladt.

Din performance vil blive bedømt som bestået eller ikke bestået.

Denne mail er en fælles mail til alle deltager, du vil straks modtage endnu en mail, med dato for hvilken dag du vil blive eksamineret. Selve eksamensdagen kommer til at forløbe som følger: Du skal møde på eksamensdagen udenfor

  Universitetsparken 1, lokale 1-0-37, kl. 9:45

og skrive dig på deltagerlisten, som vil være ophængt udenfor lokalet. Fra kl. 10:00 vil 2 lærer (Torben Mogensen, Christina Lioma, eller Jon Sporring) eksaminere i den rækkefølge, som er listen angiver, og når vi er nået igennem listen, eller for tirsdags vedkommende hvis klokken er over kl. 17:00, vil dagens eksamination være afsluttet.

Eksaminationen kommer til at forløbe som følger:

  1. Den studerende trækker et af de nedenstående 6 emner

  2. Den studerende fortæller op til 15 minuter om emnet

  3. Lærer og studerende diskuterer i ca. 10 minuter om emner fra det fulde pensum

  4. Den studerende forlader lokalet, lærerne voterer, og giver den studerende besked om resultatet.

De 6 emner, du kan trække, er:

  1. Funktionsprogrammering: Funktioner, Simple og sammensatte typer: Giv en kort beskrivelse af begreberne funktion og type, og forklar hvad forskellen mellem simple typer og sammensatte typer (compound types).  Giv eksempler på funktionsdefinitioner og forklar hvad rekursion er. Giv eksempler på alle ovenstående emner og hvordan de kan bruges sammen.

  2. Funktionsprogrammering: Lister, mængder, afbildninger (maps): Beskriv datatyperne list, endelige mængder (finite sets) og afbildninger (maps), og beskrive deres tilhørende hyppigt anvendte operatorer. Sammenlign deres ligheder og forskelle ud fra.

  3. Funktionsprogrammering: Træer (finite trees): Beskriv datatypen træer, giv eksempler på hvordan træer defineres og benyttes, hvordan funktioner der tager træer som argumenter erklæres. Giv eksempler på, hvordan binære træer kan gennemløbes og diskuter deres forskelle.

  4. Imperative programmering: Variable, sekventiel eksekvering og arrays: Beskriv forskellen mellem værdier og variable. Beskriv hvordan imperativ programmering kan gennemløbe en liste uden brug af rekursion. Diskuter forskellen mellem funktions- og imperativ programmeringsparadigmet.

  5. Objektorienteret programmering: Objektorienteret modellering: Beskriv hvordan man strukturerer en løsning af et problem ved brug af den objektorienteret designprocess. Giv et eksempel og diskuter eventuelle fordele og ulemper ved denne metode.

  6. Objektorienteret programmering: Nedarvning: Beskriv begrebet nedarvning i den objektorienteret programmerings og designmetode, og diskuter fordele og ulemper ved at strukturere et program der gør brug af nedarvning.

Held og lykke, Jon

Tilmeldingslister for Programmering og problemløsning.