Instruktion

Dette eksamenssæt indeholder 3 opgaver, der enten skal løses i C eller C++, som angivet for hver enkelt opgave.

Hver opgave vægter i den samlede karakter med de %-satser, der står ved hver af dem.

Der er i opgaverne angivet en række header- og implementeringsfiler (.c, .h, eller .cpp), der skal skrives for at løse opgaverne ved at skrive den relevante kode.

Alle disse header- og implementeringsfiler skal ved afleveringen **samles i én fil**, der afleveres **i PDF-format** på Digital Eksamen. Det skal fremgå i denne PDF-fil, hvilken opgave og fil de enkelte kode-dele kommer fra. Dvs. at kode-filer **ikke** skal afleveres enkeltvis eller som zip fil, ligesom Visual Studio solutions og projects **ikke** skal afleveres.

Husk angivelse af navn og studienummer på første side.

Der er ingen bilag.

Opgave 1 (35 %)

Denne opgave skal løses i C.

Du skal lave en funktion, der kan udskrive en række af tegnet '*' på en linie. Denne skal bruges til at tegne forskellige figurer, fx en temperaturgraf el. lign.

Funktionen skal hedde `printStars`

Den skal have én heltalsparameter.

Den skal ikke returnere nogen værdi.

a) Skriv prototypen for `printStars()`

Du skal nu implementere printStars(). Den skal selv afslutte udskriften med et linieskift.

b) Implementer `printStars()` som specificeret

`printStars()` kunne fx bruges til at udskrive følgende figur på skærmen, ved at kalde den 9 gange med forskellig parameterværdi:

```
*
**
***
****
*****
****
***
**
*
```

I alt 9 linier, hvor antallet af stjerner går fra 1 til 5 og tilbage til 1.

c) Skriv `main()` funktionen til et program, der udskriver den ønskede figur ved at kalde `printStars()` med den rigtige parameter det rigtige antal gange. Der skal bruges én eller flere programløkker.

Opgaven fortsættes på næste side!

`printStars()` skal nu bruges til at simulere en graf med resultatet fra 10 terningkast. Det er en normal terning, der kan vise fra 1 til 6 øjne, så et **eksempel** på en graf kunne være:

```
*****
****
*
***
*****
*
******
***
****
*
```

I alt 10 linier med et tilfældigt antal stjerner fra og med 1 til og med 6.

Du skal bruge `rand()` funktionen til at finde de tilfældige terningkast.

d) Udvid `main()` funktionen til programmet til at udskrive den ønskede tilfældige graf. Du skal bruge en passende programløkke, `rand()` og `printStars()`

Opgavesættet fortsættes på næste side!

Opgave 2 (20 %)

Denne opgave skal løses i C.

En funktion `makePositive` har prototypen

```
void makePositive(double * number);
```

Den tager parameteren `number` som overføres ved call-by-reference, og erstatter indholdet af den refererede variable med indholdets numeriske værdi. Dvs. hvis indholdet er negativt, erstattes det med den tilsvarende positive værdi; hvis indholdet er positivt eller 0 (nul) ændres det ikke.

a) Implementer funktionen med den beskrevne prototype og opførsel

Funktionen skal testes. Dette skal gøres ved, at der skrives et program, hvor brugeren bliver bedt om at indtaste et kommatal, kommatallet indlæses og programmet kalder ovenstående funktion og udskriver derefter indholdet af variablen, hvor kommatallet blev gemt. Forskellige værdier testes ved at brugeren kører programmet flere gange.

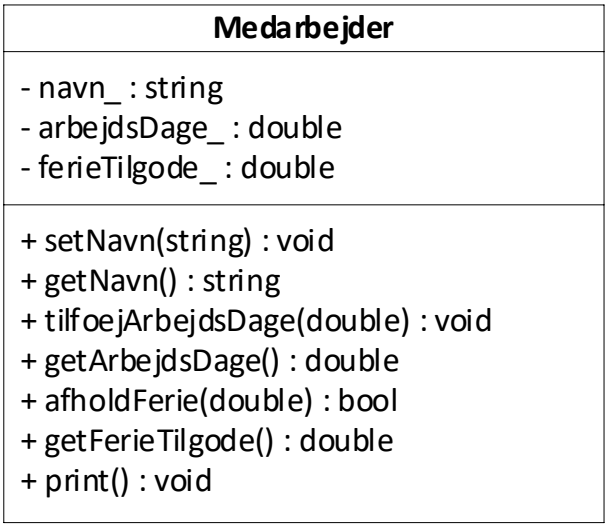
b) Skriv `main()` funktionen til et program, der implementerer testen som beskrevet

Opgavesættet fortsættes på næste side!

Opgave 3 (45 %)

Denne opgave skal løses i C++.

Denne opgave drejer sig om en klasse Medarbejder, der kan antages at være en del af et løn – og feriesystem for en virksomhed. UML klassediagrammet for klassen er angivet nedenfor:



Om klassen `Medarbejder` gælder følgende:

Attributterne skal opfylde følgende betingelser:

Navn	Beskrivelse	Gyldige værdier	Default værdi
`navn_`	Medarbejderens navn	Navnet må ikke være tomt, dvs. det skal være forskelligt fra den tomme streng `""`.	`"Ukendt"`
`arbejdsDage_`	Det samlede antal arbejdsdage, som medarbejderen har ydet, som kommatal	Må aldrig være negativt	`0`
`ferieTilgode_`	Det samlede antal feriedage, som medarbejderen kan holde ferie for, som kommatal	Må aldrig være negativt	`0`

Opgaven fortsættes på næste side!

Klassen `Medarbejder` har kun følgende parametriserede constructor.

`Medarbejder(string navn);`

Parametre:

`navn`: medarbejderens navn

Beskrivelse:

Constructoren validerer parameteren og indsætter den i attributten navn_. Hvis parameteren ikke er gyldig, indsættes defaultværdien. De øvrige attributter sættes til 0 (nul).

Om metoderne for klassen gælder:

`void tilfoejArbejdsDage(double dage);`

Parametre:

`dage`: Det antal arbejdsdage, som medarbejderen skal have ferie for

Beskrivelse:

Det checkes om parameteren `dage` er positiv. Hvis det er tilfældet, så lægger metoden parameteren dage til attributten `arbejdsDage_`.

Den udregner derefter, hvor mange feriedage værdien i parameteren giver ret til ud fra formlen

feriedage = dage * 0,1

og dette antal feriedage lægges til attributten `ferieTilgode_`.

`bool afholdFerie(double ferie);`

Parametre:

`ferie`: Det antal feriedage, som medarbejderen ønsker at holde

Returværdi:

`true`, hvis medarbejderens antal feriedage tilgode, som registreret i attributten `ferieTilgode_`, er stort nok til at afholde det ønskede antal feriedage;
`false`, hvis parameteren er negativ, eller der ikke er nok ferie tilgode

Beskrivelse:

Det checkes om parameteren `ferie` er positiv. Hvis det er tilfældet, kontrolleres det, om parameteren ferie er mindre eller lig med attributten `ferieTilgode_`. Hvis det er tilfældet, trækkes det ønskede antal feriedage fra `ferieTilgode_`, og metoden returnerer `true`. I alle andre tilfælde returnerer den `false`, og objektet ændres ikke.

Opgaven fortsættes på næste side!

`void print() const;`

Parametre:

Ingen

Beskrivelse:

Metoden udskriver medarbejderens informationer i dette pæne format, hvor de understregede data tages fra objektets tilsvarende attributter og antallet af decimaler blot er den default som udskrift til `cout` giver (du skal ikke programmere understregningerne):

```
Navn:          Hans Hansen
Arbejdsdage:   95 dage
Ferie tilgode: 9.5 dage
```

`setNavn` er en almindelig metode, der sætter den tilsvarende attribut med en valideret værdi fra parameteren ud fra ovenstående beskrivelse, og indsætter default værdien, hvis parameteren ikke indeholder en gyldig værdi.

`get`-metoderne er almindelige metoder, der bare returnerer den tilsvarende attribut.

a) Skriv hele header-filen Medarbejder.h for klassen `Medarbejder`.

b) Implementer alle metoderne ud fra specifikationen ovenfor, som de skal stå i filen Medarbejder.cpp.

Nu skal klassen testes i et program, ved at skrive en `main()` funktion i en ny fil.

Programmet skal oprette et `Medarbejder`-objekt. Programmet bruger den korrekte metode til at registrere at medarbejderne har arbejdet 22 dage. Objektet udskrives, for at beregningen af feriedage kan kontrolleres. Der skulle nu være 2,2 dages ferie til rådighed.

Derefter afholder medarbejderen 2 dages ferie, hvilket registreres ved at den korrekte metode kaldes. Objektet udskrives igen, for at den korrekte registrering af afholdt ferie kan kontrolleres. Der skulle nu være 0,2 dages ferie til rådighed.

c) Test klassen `Medarbejder` ved at skrive hele filen Opgave3.cpp med `main()` funktionen til et program, der udfører ovennævnte test.

Opgaven fortsættes på næste side!

Som en del af et lønsystem kunne man forestille sig, at programmet holder styr på de ansatte ved hjælp af et array af `Medarbejder`-objekter.

d) Udvid ovenstående test program ved at oprette et array af mindst 3 `Medarbejder`-objekter, initialiseret med forskellige navne.

e) Registrer at hver medarbejder i arrayet har arbejdet 22 dage, og udskriv objekterne. Dette skal gøres ved hjælp af en passende programløkke. Den samme løkke må gerne bruges til både at registrere og udskrive.