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. De enkelte delopgaver indgår med forskellig vægt i den samlede vurdering.

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 kodefiler **ikke** skal afleveres enkeltvis eller som zip fil, ligesom Visual Studio solutions og projects **ikke** skal afleveres. **Al kode**, der skal bedømmes, **skal stå i denne en fil**, der vil **ikke** blive kigget på kodefiler, der vedlægges som bilag.

Der skal **ikke** inkluderes screen dumps af udskrifter fra programmet i afleveringen, men det er tilladt. I så fald kan de være i samme PDF fil, eller som bilag til afleveringen. Udskrifterne fra de kørende programmer vil først og fremmest være til fordel for dig, for at tjekke, at du har løst opgaven korrekt.

Husk angivelse af navn og studienummer på første side.

Der er 2 filer med som bilag, som kan findes på Digital Eksamen, samme sted som eksamenssættet.

Opgave 1 (20 %)

Denne opgave **skal** løses i C.

Målinger af højden på mange personer har fundet en sammenhæng mellem deres opnåede højde, når de er fuldt udvoksede, og deres forældres højde.

Det kan derfor være interessant for nybagte forældre at få et gæt på, hvor høj deres nyfødte barn bliver, baseret på deres egne højder.

Sammenhængen kan sammenfattes som:

$H_{pige} = (12 * H_{far} + 13 * H_{mor}) / 26$
og
$H_{dreng} = (12 * H_{far} + 13 * H_{mor}) / 24$

Hvor H_{pige} og H_{dreng} er de forventede højder for en pige, henholdsvis en dreng, når de er fuldt udvoksede. H_{mor} og H_{far} er de faktiske højder for den biologiske mor, henholdsvis far, da de var fuldt udvoksede.

Programmet skal bede brugeren om at indtaste moderens og faderens højder i centimeter, samt om det er en dreng eller en pige. Derefter vil programmet udskrive den forventede højde på skærmen. Det er valgfrit, om der arbejdes med heltal eller kommatal. Det er valgfrit om barnets køn skal indtastes som en char, en C-string (anbefales) eller et tal. Et forløb kunne se sådan ud på skærmen:

```
Indtast moderens hoejde i cm: 175
Indtast faderens hoejde i cm: 190
Hvilket koen er barnet? (D for dreng, P for pige): D
Dit barns forventede hoejde bliver ca.: 189 cm
```

a) Skriv `main()` til et sådan program i filen Opgave1.c. Kør programmet og konstater om du har løst det rigtigt. Udskrifter fra kørsel af programmet skal ikke inkluderes i afleveringen.

Dette program kunne man jo godt have lyst til at køre flere gange, så nu skal programmet udvides, så det efter den første beregning spørger brugeren, om denne ønsker at foretage endnu en beregning. Ud fra svaret enten gentages ovenfor beskrevne forløb, eller programmet afsluttes. Det er valgfrit om svaret skal indtastes som en char, en C-string (anbefales) eller et tal.

b) Skriv denne udvidelse til `main()` i filen Opgave1.c. Kør programmet og konstater om du har løst det rigtigt. Udskrifter fra kørsel af programmet skal ikke inkluderes i afleveringen.

For en god ordens skyld tilføjes, at det drejer sig om det genetiske køn for børnene, der tages ikke højde for sjældne genetiske forekomster, det er en gennemsnitlig betragtning og programmet er mest for sjov!

Opgavesættet fortsættes på næste side!

Opgave 2 (35 %)

Denne opgave **skal** løses i C.

I statistik skal man bruge *variansen* for et datasæt, for at kunne afgøre diverse egenskaber ved disse data.

Variansen er defineret som

$$varians = \frac{\sum_{i=0}^{n-1}(x_i - g)^2}{n}$$

Hvor x_i er de enkelte talværdier i datasættet, *n* er antallet af værdier i datasættet og *g* er gennemsnittet af alle tallene i datasættet.

Dvs. man udregner summen af kvadraterne på differensen mellem hver enkelt talværdi og gennemsnittet, og dividerer summen med antallet af talværdier.

De data, denne opgave skal arbejde på, er opbevaret i et array af kommatal, hvor antallet af data i arrayet kan være forskelligt fra anvendelse til anvendelse. Det er derfor oplagt, at lave en funktion, der kan anvendes i forskellige, statistiske programmer.

Denne funktion skal hedde `varians`. Den skal have 3 parametre: et array af kommatal, størrelsen af arrayet som heltal og det forud beregnede gennemsnit som kommatal. Den skal returnere den beregnede *varians* som et kommatal.

a) Skriv prototypen for funktionen, som den skal stå i headerfilen Statistik.h

b) Implementer funktionen, som den skal stå i implementationsfilen Statistik.c

Opgaven fortsættes på næste side!

Denne funktion skal nu testes. Der skal skrives et program, der indeholder 2 arrays med nedenstående testdata, funktionen kaldes med disse, deres størrelse og det forudberegnede gennemsnit (fra tabellen), og resultatet udskrives, så det kan kontrolleres, at den fungerer korrekt.

Testdata	**Antal testdata (n)**	**Gennemsnit (g)**	**Forventet resultat fra varians()**
1 2 3 4 5 6 7 8	8	4,5	5,25
-2 -1 0 1 2	5	0	2,0

c) Skriv `main()` til et sådan program i filen Opgave2.c. Kør programmet og konstater om du har løst det rigtigt. Udskrifter fra kørsel af programmet skal ikke inkluderes i afleveringen.

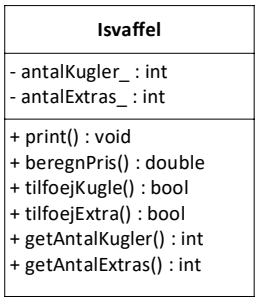
Opgavesættet fortsættes på næste side!

Opgave 3 (45 %)

Denne opgave **skal** løses i C++.

Denne opgave drejer sig om dele til et system, der kan administrere salget af isvafler fra en isbod. Isvaflerne består altid af én vaffel, mindst én kugle is og eventuelt en eller flere slags ekstra pynt eller smagsgivere. Prisen afhænger af disse bestanddele, som beskrevet nedenfor.

Her følger et klassediagram for isbodens vigtigste vare: `Isvaffel`.



Om klassen `Isvaffel` gælder følgende:

Attributterne skal opfylde følgende betingelser:

Navn	Beskrivelse	Gyldige værdier	Default værdi
`antalKugler_`	Hvor mange kugler is er der lagt i vaflen	Fra 1 til og med 5	`1`
`antalExtras_`	Hvor mange slags ekstra pynt og smagsgivere er der tilføjet til isvaflen: guf, syltetøj, flødebolle, krymmel, chili, osv	Fra 0 til og med 5	`0`

Opgaven fortsættes på næste side!

Klassen `Isvaffel` har følgende default og parametriserede constructorer.

`Isvaffel();`

Parametre:

Ingen

Beskrivelse:

Default constructor, der initialiserer de relevante attributter til default værdier.

`Isvaffel(`
`int antalKugler,`
`int antalExtras);`

Parametre:

`antalKugler`: antallet af kugler af is ved bestillingen
`antalExtras`: antallet af ekstra pynt og smagsgivere ved bestillingen

Beskrivelse:

Parametriseret constructor, der validerer parametrene, hvor det er relevant, og initialiserer de relevante attributter. Hvis en parameter ikke er gyldig, indsættes defaultværdien.

Om metoderne for klassen gælder:

`bool tilfoejKugle();`

Parametre:

Ingen

Returværdi:

`true`, hvis det var tilladt at tilføje en kugle til, ellers `false`

Beskrivelse:

Metoden tæller `antalKugler_` op med 1 hvis attributten ikke allerede er oppe på maksimum, og returnerer `true`. Hvis attributten er på maksimum allerede, kan der ikke tilføjes flere, og metoden returnerer `false`, uden at ændre attributten.

`bool tilfoejExtra();`

Parametre:

Ingen

Returværdi:

`true`, hvis det var tilladt at tilføje en extra til, ellers `false`

Beskrivelse:

Metoden tæller `antalExtras_` op med 1 hvis attributten ikke allerede er oppe på maksimum, og returnerer `true`. Hvis attributten er på maksimum allerede, kan der ikke tilføjes flere, og metoden returnerer `false`, uden at ændre attributten.

Opgaven fortsættes på næste side!

`double beregnPris();`

Parametre:
Ingen

Returværdi:
Prisen i kroner, for vaffel, iskugler og pynt

Beskrivelse:
Metoden beregner prisen for vaflen ud fra attributterne og følgende oplysninger: Vaflen koster 10 kr., hver kugle is koster 10 kr. og hver slags extra koster 5 kr.

`void print() const;`

Parametre:
Ingen

Beskrivelse:
Metoden udskriver informationer om isvaffelen i et af følgende pæne formater, alt efter om der er 1 eller flere kugler i vaflen, hvor de understregede data tages fra objektets tilsvarende attributter og metoder, og er eksempler. Understregningerne skal ikke laves i udskriften:

```
Isvaffel med 1 kugle og 2 extra ting, pris: 30 kr.

Isvaffel med 5 kugler og 3 extra ting, pris: 75 kr.
```

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

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

b) **Implementer constructorerne for `Isvaffel` og alle klassens metoder (`print()`, `beregnPris()`, `tilfoejKugle()`, `tilfoejExtra()`, `getAntalKugler()` og `getAntalExtras()`) i filen Isvaffel.cpp ud fra specifikationen ovenfor.**

Opgaven fortsættes på næste side!

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

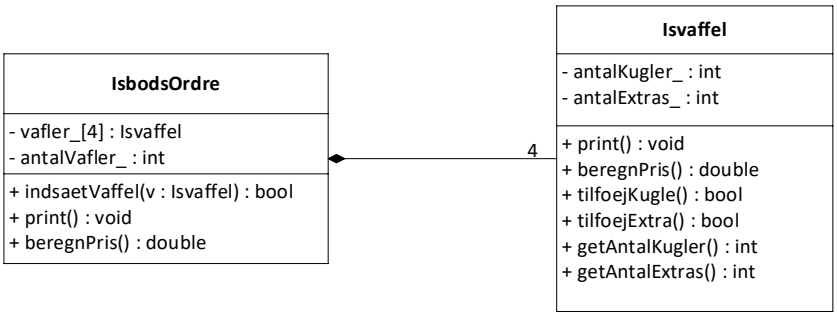
Programmet skal oprette to `Isvaffel`-objekter

- et med default constructoren
- og et med den parametriserede constructor, hvor alle parametre er forskellige fra defaultværdierne.
- Herefter udskriver programmet objekternes indhold med `print()`.
- Et af objekterne får tilføjet så mange kugler og så mange extras at maksimum ville overskrides, ved at de tilsvarende metoder kaldes et passende antal gange
- Dette objekt udskrives med `print()` så det kan konstateres at maksimummerne ikke er overskredet

c) Test klassen `Isvaffel` ved at skrive hele filen Opgave3.cpp med `main()` funktionen til et program, der udfører ovennævnte test. Kør programmet og konstater om du har løst det rigtigt. Udskrifter fra kørsel af programmet skal ikke inkluderes i afleveringen.

Opgaven fortsættes på næste side!

Klassen Isvaffel indgår i salgssystemet i isboden. Til at holde styr på en kundes ordre, med op til 4 vafler er der oprettet en klasse `IsbodsOrdre`. Relationen mellem de to klasser ses af følgende UML-diagram



`IsbodsOrdre` er defineret og delvis implementeret i filerne IsbodsOrdre.h og IsbodsOrdre.cpp, der findes som bilag, og kan indsættes i et Visual Studio projekt. En default constructor og metoderne `indsaetVaffel()` og `udregnPris()` er implementeret.

Om klassen `IsbodsOrdre` gælder følgende:

Attributterne skal opfylde følgende betingelser:

Navn	Beskrivelse	Gyldige værdier	Default værdi
`antalVafler_`	Hvor mange gyldige `Isvaffel` objekter er der indsat i ordren – altså bestilt	Fra 0 til og med 4	`0`
`vafler_[4]`	Et array af `Isvaffel` objekter, hvor hver af dem er tilføjet med metoden `indsaetVaffel`, hvor det første indsættes med index 0, osv.	Gyldige `Isvaffel` objekter	Default `Isvaffel` objekter.

`IsbodsOrdre();`

Parametre:

Ingen

Beskrivelse:

Default constructor, der initialiserer de relevante attributter til default værdier.
Bemærk, denne er allerede implementeret!

Opgaven fortsættes på næste side!

Om metoderne for klassen gælder:

`void indsaetVaffel(Isvaffel v);`

Parametre:

`v: Isvaffel` objekt.

Beskrivelse:

Metoden kopierer `Isvaffel` objektet `v` ind på den næste ledige plads i `vafler_`, hvis der ikke allerede er indsat 4 isvafler, dvs `antalVafler_` skal være mindre end 4. Den tæller så `antalVafler_` op med 1 hvis attributten ikke allerede er oppe på maksimum. Hvis der allerede er 4 isvafler i ordren, gør metoden ikke noget.

Bemærk, denne er allerede implementeret!

`double beregnPris();`

Parametre:

Ingen

Returværdi:

Prisen i kroner, beregnet som summen af alle `Isvaffel` objekter, der er indsat.

Beskrivelse:

Metoden beregner den samlede pris for ordren, ved at udregne summen af prisen for alle de indsatte `Isvaffel` objekter.

Bemærk, denne er allerede implementeret!

`void print() const;`

Parametre:

Ingen

Beskrivelse:

Metoden udskriver information om hvor mange isvafler, der er i ordren, om hver enkelt isvaffel og den totale pris i følgende pæne format hvor de understregede data tages fra objektets tilsvarende attributter og metoder, og er eksempler. Understregningerne skal ikke laves i udskriften:

```
Ordre paa 2 vafler
Isvaffel med 5 kugler og 5 extra ting, pris: 85 kr.
Isvaffel med 2 kugler og 3 extra ting, pris: 45 kr.
Pris ialt: 130 kr.
```

d) Implementer metoden `print()` for `IsbodsOrdre` i filen IsbodsOrder.cpp, som skal udskrive alle de vafler, der er indsat i ordren og udregne og udskrive den totale pris, på samme måde som ovenstående eksempel.

e) Opret et `IsbodsOrdre` objekt i den tidligere oprettede `main()`, indsæt de to `Isvaffel` objekter og udskriv ordren. Udskrifter fra kørsel af programmet skal ikke inkluderes i afleveringen.