UNIVERSITETET I AGDER

FAKULTET FOR TEKNOLOGI OG REALFAG

E K S A M E N

Emnekode: **MAS234**
Emnenavn: **Innebygde datasystemer for mekatronikk**

Dato: 11. desember 2018
Varighet: 4 timer

Antall sider inkl. forside: 10

Tillatte hjelpemidler: Alle trykte og håndskrevne hjelpemidler tillatt. Kalkulator tillatt.

Merknader:

- Kandidaten må selv kontrollere at oppgavesettet er fullstendig.
- Les nøye igjennom oppgavene slik at du forstår hva det spørres etter.
- Beskriv eventuelle antagelser du må gjøre dersom oppgaven er formulert uklart. Lesing av spesifikasjoner, herunder oppgavetekster, er en del av det dere blir testet i.
- All kildekode i oppgaveteksten er gitt i C++11. Der det spørres om implementasjon i en eller annen form, skal det programmeres C++ på papir.
- Det kan antas at header <iostream> er inkludert fra før, alle andre nødvendige includes må spesifiseres i løsningen.
- Kodesnuttene i oppgaveteksten er i en del tilfeller listet "frittstående". Det skal antas at disse kjøres i kontekst av en funksjon (eksempelvis inni en main-funksjon).

Oppgave 1 (7%)

Programforståelse.

Gitt følgende kodesnutt:

```
const int x = 3;
const int y = 2;
int z = 82;

const int a = x/y;
int* b = &z;

std::cout << "a = " << a << ", b = " << *b << std::endl;
```

a) Kodesnutten er plassert inni en main-funksjon i en cpp-fil. Hvis vi antar at cpp-filen kompileres og programmet kjøres: Hva skrives ut?

a = 1, b = 82

b) Hva betyr * på linje 6 i kodesnutten?

Det betyr at vi deklarerer en peker. I dette tilfellet en peker til en `int`.

c) Hvilken funksjon har * slik operatoren benyttes på siste linje i kodesnutten?

* er her dereference-operator, og anvendes på en pekervariabel for å hente ut verdien på pekeradressen.

d) Kodesnutten er plassert inni en main-funksjon og er opprinnelig laget for bruk på en PC med MS Windows 10. Du bestemmer deg for å skrive om programmet for bruk med Arduino. Hvor plasserer du kodesnutten nå, hvis den fremdeles skal kjøres én gang? Og må noe annet endres?

Det er ikke nødvendig å skrive om programmet for Arduino for å løse denne deloppgaven.

Koden kan eksempelvis plasseres i setup-funksjonen.

Det må gjøres endringer med tanke på å håndtere utskrift til terminal. Dersom serieport skal anvendes på Arduino-en, så kan det enten benyttes innebygget bibliotek med printf, eller det kan anvendes en kompakt versjon av iostream skrevet for Arduino.

Oppgave 2 (7%)

Funksjoner.

a) Skriv deklarasjonen til en funksjon som tar inn to heltall av valgfri heltallstype og returnerer en bool. Funksjonen skal hete isGreater.

```
bool isGreater(uint8_t first, uint8_t second);
```

b) Lag implementasjonen til funksjonen fra deloppgave a). Funksjonen skal returnere true hvis første argument er større enn andre argument, false ellers.

```
bool isGreater(uint8_t first, uint8_t second)
{
    return first > second;
}
```

Oppgave 3 (15%)

Løkker.

Følgende kodesnutt printer tall til standard output (vanligvis terminalvinduet).

```
const uint8_t n = 25;

for (uint8_t i = 0; i <= n; i++)
{
    std::cout << +i << std::endl;
}
```

a) Hva er første og siste tall som skrives ut?

Henholdsvis 0 og 25.

b) Hvor mange tall skrives ut?

26

c) Hvorfor bør variabelen ”n” være const i denne kodesnutten?

Fordi det ikke er behov for å endre på verdien etter initialisering.

d) Skriv en tilsvarende implementasjon som skriver ut hvert fjerde tall, fra og med 4, til og med 28.

```
for (uint8_t i = 4; i <= 28; i+=4)
{
    std::cout << +i << std::endl;
}
```

Oppgave 4 (8%)

Programforståelse.

Følgende swap-funksjon er skrevet for å bytte verdi mellom to variabler. Main-funksjonen er et eksempel på bruk av funksjonen.

```
void swap(int first, int second)
{
    const int temp = first;
    first = second;
    second = temp;
}

int main()
{
    int x = 314;
    int y = -2300;

    swap(x, y);

    std::cout << "x=" << x << " y=" << y << std::endl;
    return 0;
}
```

a) Vil programmet kompilere, og er det kjørbart? Hva printes ut hvis det kan kjøres?

Programmet lar seg kompilere, og er kjørbart. Utskrift til terminal:

x=314 y=-2300

b) Fungerer programmet i henhold til spesifikasjonen (altså bytter det verdien på variablene)? Hvis ikke, foreslå en endring ved å skrive en ny og fungerende swap-funksjon.

Nei, det bytter ikke om på verdiene. En mulig endring er å anvende referanser. Bruk av pekere på rett måte kan også være en god løsning her.

```
void swap(int& first, int& second)
{
    const int temp = first;
    first = second;
    second = temp;
}
```

c) Bør variablene x og y i main deklareres const? Hvorfor / hvorfor ikke?

Nei, bytte av verdi krever at vi skriver til variablene.

Oppgave 5 – Objektorientering (15%)

Gitt følgende C++-klasser:

```
class SensorData
{
public:
    SensorData(long t, double v)
        : timeStamp_us(t)
        , value(v)
    {
    }

    long timeStamp_us;
    double value;
};



class ISensorDataReceiver
{
public:
    virtual ~ISensorDataReceiver() { }
    virtual void setSensorData(SensorData& sd) = 0;
};



class TemperatureReceiver : public ISensorDataReceiver
{
public:
    TemperatureReceiver() { }

    virtual ~TemperatureReceiver() { }

    virtual void setSensorData(SensorData& sd)
    {
        if (timeStamp_us_ > sd.timeStamp_us)
        {
            timeStamp_us_ = sd.timeStamp_us;
            temperature_ = sd.value;
        }
    }

private:
    long timeStamp_us_;
    double temperature_;
};
```

Oppgave 5 - fortsettelse fra forrige side:

a) Skriv implementasjonen til en main-funksjon som:

I. Først oppretter et SensorData-objekt med tidsstempel 12345 og måleverdi 38.9

II. .. deretter oppretter et TemperatureReceiver-objekt,

III. .. og til slutt sender inn det opprettede SensorData-objektet til TemperatureReciver-instansen ved hjelp av setSensorData-funksjonen.

```
int main()
{
    SensorData sd(12345l, 38.9); // I

    TemperatureReceiver tr;      // II

    tr.setSensorData(sd);        // III

    return 0;
}
```

b) Hvorfor kan det ikke opprettes en instans av ISensorDataReceiver-klassen, og hva kaller vi denne typen klasser i C++ ?

(ISensorDataReceiver er et interface, og dette lages i C++11 ved bruk av typen klasse det spørres om).

Klassen ISensorDataReceiver har en pure virtual medlemsfunksjon (=0), og er dermed en abstrakt klasse. Abstrakte klasser kan ikke instansieres.

c) Formålet med sjekken i setSensorData-funksjonen er å sikre at vi kun tar i mot målinger som er nyere enn den forrige vi mottok.

Det har sneket seg inn en liten feil i setSensorData-funksjonen. Hva er feilen?

Det kreves at nytt tidsstempel er mindre enn eksisterende tidsstempel, noe som neppe gir mening. Det korrekte ville vært å kreve at tidsstempel på ny måling er større enn tidsstempelet på forrige måling:

```
if (sd.timeStamp_us > timeStamp_us_)
```

Oppgave 6 (8%)

Udefinert oppførsel.

a) Hva er udefinert oppførsel, slik det er definert i C++-standarden?

Udefinert oppførsel opptrer når gitte språklige regler i C++ ikke overholdes. Konsekvensen er at programmet kan gjøre «hva som helst», og vi har dermed et meningsløst og potensielt farlig program

b) Hvorfor er det viktig å unngå å skrive programmer hvor udefinert oppførsel forekommer?

Udefinert oppførsel kan føre til at programmet krasjer, påvirker andre programmer eller operativsystemet, endrer data i feil del av programmet etc. Det kan være vanskelig å detektere denne typen feil under testing, og det kan også gjøre programmet mindre porterbart da det ikke følger C++-standarden. Feil kan komme til overflaten når tilsynelatende urelaterte endringer innføres.

c) Gi et eksempel på udefinert oppførsel i C++.

Flere mulige svar. F.eks. integer overflow (signed integer), bruk av peker til objekt som er deleted, aksessere element etter siste eksisterende element i en array etc.

Oppgave 7 (5%)

Datatypen struct.

a) Lag en struct som inneholder feltene sensorId, timeOfValidity og measuredTemperature. Velg passende datatyper, og begrunn valgene kort.

Struct-en skal kun deklareres.

```
struct TemperatureMeasurement
{
    uint8_t sensorId;
    long timeOfValidity;
    float measuredTemperature;
};
```

b) Hva skiller en C++-struct fra en C++-klasse?

Standard synlighet på medlemsvariable og medlemsfunksjoner er public i en struct og private for en class (§ 11.2 i C++11-standarden).

Oppgave 8 (10%)

Begrepsforståelse.

a) Beskriv kort hva som skjer når vi kompilerer et C++-program.

Menneskelig lesbar C++-kode blir omformet assembly-kode. Dette skjer etter preprosessering, men før assembler og linking til kjørbar fil (hex-fil, elf-fil, a.out etc..). Det gis halv score på å omtale hele prosessen som kompilering.

(Det teller positivt med en figur som viser stegene med preprosessering, kompilering, assembler og linking men det er ikke et krav om figur i besvarelsen for full uttelling.)

b) Hva skiller "ordinær" kompilering fra krysskompilering?

Ved krysskompilering kompileres det for enn annen plattform enn det host-maskinen (maskinen som kompilerer) kjører på. Eksempelvis kryskompilering til en 8 bit Atmega128 på en Windows 10-maskin med x86-64-prosessor.

c) Under mikrokontrollerlab-en med AVR mikrokontrollere, ble det benyttet en Atmel ICE for skriving til flash-minnet på mikrokontrolleren (med mer).

Skjer skriving av flash-minnet på mikrokontrolleren før eller etter krysskompilering?

Etter.

Oppgave 9 (10%)

Mikrokontrollerkretser

a) Hvilken funksjon har typisk en spenningsregulator i en mikrokontrollerkrets?

Den skal forsyne kretsen med stabil og mest mulig støyfri spenning på et gitt nivå.

b) Nevn minst to typer spenningsregulatorer, og forklar hva som skiller disse.

Linear (f.eks. LDO) og switched mode (f.eks. buck).

En lineær spenningsregulator gir typisk en stabil og fin utspenning med lite støy. Imidlertid medfører den stort effekttap i form av varme dersom inn-spenningen er mye større enn ut-spenningen og strømmen er stor. Spenningsfallet blir direkte et tap som funksjon av strømmen.

En switched-mode strømforsyning (DC/DC) av typen buck er en step-down-konverter, og kan på samme måte som en Lineær spenningsregulator kun regulere en innspenning ned til et lavere nivå. Buck-konverteren er basert på at en spole og en kondensator tilføres energi når en transistor slås på, og deretter lades ut når transistoren er av (f.eks. gjennom en diode eller en ekstra transistor). Dette gjøres høgfrekvent, slik at lasten kan tilføres en lavere spenning med rimelig lite rippel. Imidlertid støyer switched mode strømforsyninger generelt mer enn en lineær strømforsyning. Den store fordelen til en switched mode spenningsregulator er at den i de fleste tilfeller vil ha et langt lavere tap enn en lineær spenningsregulator når innspenningen er høg sammenlignet med utspenningen.

c) En rød LED med spenningsdropp $V_f = 2.3\ V$ skal styres av en mikrokontroller med supplyspenning $V_s = 5\ V$. Tegn et enkelt kretsskjema for hvordan dette bør kobles. Skriv på komponentverdier, og vis hvordan du kommer frem til disse.

Det er ønskelig med maksimal lysstyrke, i og med at anvendelsen er utendørs. Komponentenes levetid er imidlertid også viktig.

<u>Se Vedlegg A for utsnitt fra LED-ens datablad</u>.

Tegningen skal inneholde motstand i serie med LED i rett retning sammenlignet med supplyspenningen, og koblet til en pinne på mikrokontrolleren. Det bør også tas en

vurdering på om pinnen på mikrokontrolleren kan source eller sinke nødvendig strøm.

Verdi på motstanden bør være: $R_{min} = \frac{V_s - V_f}{I_{max}} = 90\ \Omega$.

c) Hvis mikrokontrolleren i deloppgave c) er av AVR-type; hvilke steg er nødvendige for å kunne styre den programmatisk når du har koblet opp i henhold til kretsskjemaet ditt fra c) ? (Hva må du gjøre i f.eks. et C++-program for å styre LED-en)?

Rett pinne på rett port må settes som utgang. Deretter på det skrives lav til pinnen for å få lys ved sinking eller høg ved sourcing.

Oppgave 10 (5%)

Nettverkstopologier.

Hvilke typer (én eller flere) nettverkstopologier er støttet med CAN? Tegn ett eller flere eksempler.

Bus (point-to-point kan også aksepteres, men kun i tillegg til bus).

Oppgave 11 (5%)

Operativsystemer I.

Forklar kort:

1) Hva de viktigste oppgavene til et operativsystem er,

   Ikke-prioritert rekkefølge:
   - Ressurshåndtering; fordeling og tilgangsstyring. Scheduling av prosesser, håndtering av systemminne og styre og aksessere IO-ressurser.
   - Tilrettelegge for videreutvikling. Eksempelvis ved å håndtere hardware-abstraksjon, tilby tjenester, være vedlikeholdbare (system for patching og oppgradering av OS og gjerne også applikasjoner).
   - Bekvemmelighet. Gi applikasjonsprogrammereren et i stor grad programmeringsspråk- og hardware-uavhengig ABI og API. ABI utnyttes for å kunne la programmer skrevet i ulike språk samhandle (gjennom funksjonskall), og API-er lar programmereren utnytte (i stor grad) hardware-uavhengige høgnivå-grensesnitt for interaksjon med operativsystemet og derigjennom også maskinvareressurser. Eksempelvis bruk av innebygget støtte for nettverkskommunikasjon (ethernet, CAN etc.).

2) hvilke egenskaper i operativsystemet som er mest sentrale når det skal anvendes som kjøremiljø for styring av tidskritiske fysiske systemer. (Eksempelvis styring av et fysisk system med rask dynamikk).

   Sanntidsegenskaper. IO-støtte (tilgjengelige drivere for nødvendig hardware).

Oppgave 12 (5%)

Operativsystemer II.

Foreslå et passende operativsystem for hvert av de følgende tilfellene. Gi en 1-2 setningers begrunnelse for hvert svar.

a) Utvikling av desktop-applikasjon for registrering av måledata i et laboratorium.

   MS Windows, MacOS eller Ubuntu Linux. Det viktige her er å ha tilgjengelig gode rammeverk for datainnsamling og presentasjon (GUI, biblioteker for formatstøtte

etc.)

b) Lukket sløyfe-styring av en selvbalanserende robot. (Inkludert indre sløyfer.)

Her vil det være naturlig å velge et operativsystem med gode sanntidsegenskaper, eksempelvis Linux-variant med sanntids-patch, QNX, VxWorks eller lignende.

Vedlegg A – Utsnitt fra datablad

SMD LED

Red 2.1...2.6 V 0603

RND components

SPECIFICATION:

Package	0603
Light colour	Red
Transmission angle	130 °
Case colour	Transparent
Housing type	SMD
Conducting-state voltage max.	2.6 V
Conducting-state voltage min.	2.1 V
Dominant wavelength	631 nm
Forward current	30 mA
Forward voltage	2.1...2.6 V
Luminous intensity	80 mcd
Luminous intensity max.	80 mcd
Luminous intensity min.	80 mcd
Off-state current	10 µA
Off-state voltage	5 V
Operating temperature	-40...+85 °C
Peak wave length	645 nm
Power	78 mW
Technology	AlGaInP