

SENSORVEILEDNING

For eksamen i: **INF-1100**

Dato: **Torsdag 25. februar 2015**

Sensorveiledningen er på 7 sider inklusiv forside

Fagperson/intern sensor: Robert Pettersen

Telefon: 47247552

**Eksamen INF-1100
Innføring i programmering og
datamaskiners virkemåte
Vår 2016**

Eksamenssettet består av 4 oppgaver.

Der oppgaven ber om at du skriver en funksjon kan du bruke C lignende pseudo-kode. Husk også at du kan referere tilbake til funksjoner du tidligere har definert.

Oppgave 1 - 20%

Gi en kort beskrivelse av von Neumann modellen.

Besvarelsen bør beskrive de fem komponentene (processing unit, control unit, memory, input og output) og interaksjonen mellom disse.

Dersom beskrivelsen er god nok bør studenten få full uttelling.

Oppgave 2 - 20%

Hvilken verdi vil funksjonen *ukjent* nedenfor returnere ved følgende kall:

a) *ukjent*(2, 1)

Funksjonen utfører multiplikasjon.

Løsningsforslag 1a:

2

b) *ukjent*(5, 3)

Løsningsforslag 1b:

15

```
int ukjent(int a, int b)
{
    int p;

    p = 0;
    while (b != 0) {
        if ((b & 0x1) != 0) {
            p = p + a;
        }
        a = a << 1;
        b = b >> 1;
    }
    return p;
}
```

Oppgave 3 - 25%

Gitt at en tekststreng er et array med elementer av type *char* og hvor siste element i arrayet har verdien 0.

- a) Skriv en funksjon *strlengde* avgjør antall elementer i en tekststreng, inkludert elementet med verdien 0:

```
int strlengde(char *s)
```

Løsningsforslag 3a:

```
int strlengde(char *s)
{
    int len;
    for (len = 0; *s != 0; s++)
        len++;
    return len;
}
```

- b) Skriv en funksjon *strsiste* som returnerer arrayposisjonen til siste forekomst av et element *c* i en tekststreng *s*:

```
int strsiste(char *s, char c)
```

strsiste skal returnere verdien -1 dersom *c* ikke forekommer i *s*.

Løsningsforslag 3b:

```
int strsiste(char *s, char c)
{
    int pos;
    int siste;

    siste = -1;
    for (pos = 0; s[pos] != 0; pos++) {
        if (s[pos] == c)
            siste = pos;
    }
    return siste;
}
```

- c) Skriv en funksjon *strforekommer* som avgjør om en tekststreng *b* forekommer i en tekststreng *a*:

```
int strforekommer(char *a, char *b)
```

strforekommer skal returnere 1 dersom *b* forekommer i *a*. -1 skal returneres dersom *b* ikke forekommer i *a*. For eksempel, tekststrengen “over” forekommer i tekststrengen “avisoverskrifter”, men “gen” forekommer ikke i “formel”.

Løsningsforslag 3c:

```
int match(char *a, char *b)
{
    int i;

    for (i = 0; a[i] != 0 && b[i] != 0 && a[i] == b[i]; i++) ;

    if (b[i] == 0)
        return 0;
    return -1;
}

int strforekommer(char *a, char *b)
{
    int i;

    for (i = 0; a[i] != 0; i++) {
        if (match(a + i, b) == 0)
            return 1;
    }
    return -1;
}
```

Oppgave 4 - 35%

Denne oppgaven involverer bruk av lister og et angitt sett med listefunksjoner. Bruk de angitte listefunksjonene i besvarelsen. **Ikke** gjør antagelser om hvordan listene er implementert.

- a) Skriv en funksjon som avgjør om en liste inneholder et bestemt element:

```
int list_contains(list_t *list, void *item)
```

list_contains skal returnere 1 dersom det angitte elementet (*item*) eksisterer i listen (*list*) og 0 dersom det ikke eksisterer. Her må du bruke listeiteratorene. Du kan anta at det eksisterer en funksjon *isequal* som avgjør om to elementer er like. *isequal* returnerer 1 dersom de to angitte elementene er like og 0 dersom de ikke er like:

```
int isequal(void *itemX, void *itemY)
```

Løsningsforslag 4a:

```
int list_contains(list_t *list, void *item)
{
    list_iterator_t *iter;
    void *tmp;

    iter = list_createiterator(list);
    tmp = list_next(iter);
    while (tmp != NULL) {
        if (isequal(tmp, item) == 1)
            break;
        tmp = list_next(iter);
    }
    list_destroyiterator(iter);
    if (tmp != NULL)
        return 1;
    else
        return 0;
}
```

- b) Gitt en liste *A* og en liste *B*, skriv en funksjon som konstruerer en ny liste med de elementer som eksisterer i både *A* og *B*:

```
list_t *list_containsboth(list_t *A, list_t *B)
```

Her kan du gjenbruke funksjonen *list_contains* fra forrige oppgave.

Løsningsforslag 4b:

```
list_t *list_containsboth(list_t *A, list_t *B)
{
    list_t *new;
    list_iterator_t *iter;
    void *item;

    new = list_create();
    iter = list_createiterator(A);
    item = list_next(iter);
    while (item != NULL) {
        if (list_contains(B, item) == 1) {
            list_addfirst(new, item);
        }
        item = list_next(iter);
    }
    list_destroyiterator(iter);
    return new;
}
```

Du kan anta at følgende listefunksjoner er tilgjengelige.

```
// Lag en ny liste
list_t *list_create(void);

// Sett inn et element først i en liste
int list_addfirst(list_t *list, void *item);

// Lag en ny listeiterator
list_iterator_t *list_createiterator(list_t *list);

// Returner element som pekes på av iterator og
// la iterator peke på neste element. NULL returneres
// når det ikke finnes noe neste element.
void *list_next(list_iterator_t *iter);

// Frigi iterator
void list_destroyiterator(list_iterator_t *iter);
```