

Eksamen INF-1100
Innføring i programmering og
datamaskiners virkemåte
Høst 2014

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 - 25%

Anta at symbolet **&** representerer en *bitvis and* operasjon mellom to operander. Følgende tabell viser resultatet av **&** operasjonen mellom to binære siffer:

```
0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1
```

Anta at symbolet **!** representerer en *logisk not* operasjon på en operand. En **!** operasjon resulterer i verdien 1 dersom operanden har verdien 0. En **!** operasjon resulterer i verdien 0 dersom operanden har en verdi ulik 0.

Gitt følgende funksjon:

```
int ukjent(int x)
{
    return !(x & (x-1));
}
```

- a) Hva vil funksjonen *ukjent* returnere dersom x har verdien 4? Vis hvordan du har kommet frem til svaret.

Løsningsforslag 2a) 1.

```
4 & 3 = 100b & 011b = 000b
!000b = 1
```

- b) Hva vil funksjonen *ukjent* returnere dersom x har verdien 6? Vis hvordan du har kommet frem til svaret.

Løsningsforslag 2b) 0.

```
6 & 5 = 110b & 101b = 100b
!100b = 0
```

- c) Hva vil funksjonen *ukjent* returnere dersom x har verdien 8? Vis hvordan du har kommet frem til svaret.

Løsningsforslag 2c) 1.

```
8 & 7 = 1000b & 0111b = 0000b
!0000b = 1
```

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 avgjør 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 “eksamen” forekommer i tekststrengen “eksamensforberedelse”, men “is” forekommer ikke i “nordlys”.

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

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

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

Oppgave 4 - 30%

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.

Anta at ordene i et tekstdokument er satt inn i en liste.

- a) Skriv en funksjon *antallord* som avgjør hvor mange ord dokumentet består av:

```
int antallord(list_t *tekst)
```

Løsningsforslag 4a)

```
int antallord(list_t *tekst)
{
    int antall;
    list_iterator_t *iter;

    antall = 0;
    iter = list_createiterator(tekst);
    while (list_next(iter) != NULL)
        antall++;

    list_destroyiterator(iter);
    return antall;
}
```

- b) Anta at en ønsker å finne ord i dokumentet som er feilstavet. Denne oppgaven kan løses ved å sjekke om ordene i dokumentet forekommer i en ordliste. Dersom et ord ikke forekommer i ordlisten kan du anta at ordet er feilstavet. Anta at en slik ordliste er tilgjengelig og skriv en funksjon som returnerer en liste med alle ord som er feilstavet.

```
list_t *feilstavet(list_t *tekst, list_t *ordliste)
```

Du kan anta at det eksisterer en funksjon *strcmp(char *a, char *b)* som kan benyttes for å sammenligne ord. *strcmp* returnerer verdien 0 kun dersom tekststrengene *a* og *b* er identiske.

Løsningsforslag 4b)

```
int contains(list_t *ordliste, char *ord)
{
    list_iterator_t *iter;
    char            *word;

    iter = list_createiterator(ordliste);
    word = list_next(iter);
    while (word != NULL) {
        if (strcmp(word, ord) == 0)
            break;
        word = list_next(iter);
    }
    list_destroyiterator(iter);

    if (word != NULL)
        return 0;
    return -1;
}

list_t *feilstavet(list_t *tekst, list_t *ordliste)
{
    list_t      *new;
    list_iterator_t *iter;
    char        *ord;

    new = list_create();
    iter = list_createiterator(tekst);
    ord = list_next(iter);
    while (ord != NULL) {
        if (contains(ordliste, ord) == -1)
            list_addfirst(new, ord);

        ord = list_next(iter);
    }
    list_destroyiterator(iter);
    return new;
}
```

- c) Anta at en ønsker å finne ut om to tekstdokumenter er identiske. Skriv en funksjon som sammenligner to lister for å finne ut om de er identiske:

```
int sammenlign(list_t *tekstA, list_t *tekstB)
```

sammenlign skal returnere 0 dersom *tekstA* og *tekstB* inneholder samme ord i samme rekkefølge. *sammenlign* skal returnere -1 dersom dokumentene ikke er identiske. Du kan anta at *strcmp* funksjonen beskrevet ovenfor i **b)** eksisterer.

Løsningsforslag 4c)

```
int sammenlign(list_t *tekstA, list_t *tekstB)
{
    list_iterator_t *iterA, *iterB;
    char            *ordA, *ordB;

    iterA = list_createiterator(tekstA);
    iterB = list_createiterator(tekstB);
    ordA = list_next(iterA);
    ordB = list_next(iterB);

    while (ordA != NULL && ordB != NULL) {
        if (strcmp(ordA, ordB) != 0)
            break;
        ordA = list_next(iterA);
        ordB = list_next(iterB);
    }
    list_destroyiterator(iterA);
    list_destroyiterator(iterB);

    if (ordA == NULL && ordB == NULL)
        return 0;
    return -1;
}
```

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);

// Frigi liste
void list_create(list_t *list);
```

```
// 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);
```