

# LØSNINGSFORSLAG

---

For eksamen i:      **INF-1100 - Innføring i programmering  
og datamaskiners virkemåte**

Dato:                      **Tirsdag 8. desember 2015**

## Oppgave 1

a)

A	B	C	X	Y
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

- b) Den logiske kretsen beregner  $A + B + C$ , med to output bits. X er den mest signifikante biten i resultatet og Y er den minst signifikante biten i resultatet. Kretsen kan også brukes som ei byggeblokk i en større adder, hvor A og B er input bits, C er carry input, Y er output bit, og X er carry output.

## Oppgave 2

- a) Et register i en prosessor er et lager som kan aksesseres raskere og på en mer direkte måte enn det vanlige minnet. Registerne kan være input og output for instruksjoner.
- b) PC brukes til å lagre minneadressen til neste instruksjon som skal utføres. Den endrer verdi for hver instruksjon som utføres. Enten fordi instruksjonen eksplisitt endrer PC, i form av et ubetinget eller betinget hopp (jump eller branch) eller fordi PC implisitt skal økes til å peke på den etterfølgende instruksjonen.
- c) MAR og MDR brukes i kombinasjon med kontrollsignaler til å kommunisere med minnet. MAR inneholder en minneadresse som skal leses eller skrives, og MDR vil enten inneholde verdien som ble lest, eller verdien som skal skrives.

### Oppgave 3

```
int gcd(int a, int b)
{
    while (b != 0) {
        int t = b;
        b = a % b;
        a = t;
    }
    return a;
}
```

### Oppgave 4

```
void reducer(R *r)
{
    if (r->nevner < 0) {
        r->teller *= -1;
        r->nevner *= -1;
    }
    int g = gcd(r->teller, r->nevner);
    r->teller /= g;
    r->nevner /= g;
}
```

### Oppgave 5

```
void add(R *r, R *x, R *y)
{
    int tellerA = x->teller * y->nevner;
    int tellerB = y->teller * x->nevner;
    r->teller = tellerA + tellerB;
    r->nevner = x->nevner * y->nevner;
    reducer(r);
}

void mul(R *r, R *x, R *y)
{
    r->teller = x->teller * y->teller;
    r->nevner = x->nevner * y->nevner;
    reducer(r);
}
```

## Oppgave 6

```
void sub(R *r, R *x, R *y)
{
    R minusY = { -y->teller, y->nevner };
    add(r, x, &minusY);
}

void div(R *r, R *x, R *y)
{
    R inversY = { y->nevner, y->teller };
    mul(r, x, &inversY);
}
```

## Oppgave 7

```
int lengde(node_t *liste)
{
    int n = 0;
    while (liste != NULL) {
        liste = liste->neste;
        n += 1;
    }
    return n;
}

void sum(R *r, node_t *liste)
{
    if (liste != NULL) {
        r->teller = 0;
        r->nevner = 1;
        while (liste != NULL) {
            add(r, r, &liste->verdi);
            liste = liste->neste;
        }
    }
}

void gjennomsnitt(R *r, node_t *liste)
{
    if (liste != NULL) {
        R ntedel = {1, lengde(liste)};
        sum(r, liste);
        mul(r, r, &ntedel);
    }
}
```

```

int sammenlign(R *a, R *b)
{
    int x = a->teller * b->nevner;
    int y = b->teller * a->nevner;
    if (x < y)
        return -1;
    if (x > y)
        return 1;
    return 0;
}

void minimum(R *r, node_t *liste)
{
    if (liste != NULL) {
        R *min = &liste->verdi;
        liste = liste->neste;
        while (liste != NULL) {
            if (sammenlign(&liste->verdi, min) < 0)
                min = &liste->verdi;
            liste = liste->neste;
        }
        r->teller = min->teller;
        r->nevner = min->nevner;
    }
}

```