Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4105 IT Grunnkurs
Høst 2014

Løsningsforslag — Øving 7

1 Teori

a) Binært: 01000101, Heksadesimalt: 45

b)
1. Paritet
2. Sjekksum
3. Hash-funksjon

c) En algoritme er en presis beskrivelse av en endelig serie operasjoner som skal utføres for å løse et problem. (Kilde: Wikipedia)

2 Kodeforståelse

a) Funksjonen roterer verdiene i en vektor ett hakk til venstre. For eksempel:
[1 2 3] => [2 3 1]

b) Funksjonen reverserer en streng, f.eks returnerer `unknown2('teststreng')` strengen `'gnerstset'`.

3 Massemidtpunkt

a)

```
function center = center_of_mass( line )

    total_weight = sum(line);
    halfway = total_weight / 2;
    weight_so_far = 0;

    i = 1;
    while (weight_so_far + line(i)) <= halfway

        weight_so_far = weight_so_far + line(i);
        i = i + 1;

    end
    center = i-1;

    center = center + (halfway - weight_so_far) / line(i);
end
```

b)

```
line = rand(1, 13)*100;
center_of_mass(line)
```

4 Omkrets

a)

```
function res = pytagoras( a, b )
    res = sqrt( a^2 + b^2 );
end
```

```
function length = perimeter( x, y )

    length = 0;

    n = size(x, 2);
    if n > 0 && size(x, 2) == size(y, 2)
        for i = 1:n-1

            a = x(i) - x(i+1);
            b = y(i) - y(i+1);

            length = length + pytagoras( a, b );
        end

        a = x(n) - x(1);
        b = y(n) - y(1);

        length = length + pytagoras( a, b );
    end
end
```

5 Strenghåndtering

a)

```
function res = isPalindrome(str)
  n = size(str, 2);

  res = true;
  for i=1:n
    if str(i) ~= str(n+1-i)
      res = false;
      break
    end
  end
end
```

b)

```
function pos = findsub(needle, haystack)

    n = size(needle, 2);
    N = size(haystack, 2);
    pos = -1;

    for i = 1:N-n+1
       substring = haystack(i:i+n-1);
       if equal(substring, needle)
           pos = i;
           return;
       end
    end
end
```

6 Investeringsstrategier

a)

```
closing_prices = [100 101 102 100 102 104 103 98 96 101];
```

b)

```
plot(1:10, closing_prices);
```

c)

```
function dailyReturn = daily_returns( closing_price )

n = size(closing_price, 2);
dailyReturn = zeros(n, 1);

for i = 2:n
  dailyReturn(i) = closing_price(i) - closing_price(i-1);
end

end
```

d)

```
plot(1:10, daily_returns( closing_prices );
```

e)

```
function res = is_going_up( n, i, daily_return )

  res = true;

  for j = i:-1:max(1, i-n+1)
    if (daily_return(j) < 0)
      res = false;
      return;
    end
  end

end
```

f)

```
function returns = momentum( start_amount, n, closing_price )

  daily_return = daily_returns(closing_price);
  cash = start_amount;
  invested = 0;

  for i = n:size(closing_price, 2)

    invested = invested * closing_price(i) / closing_price(i-1);

    if (is_going_up(n, i, daily_return))
      % Buy
      invested = invested + cash;
      cash = 0;
    else
      % Sell
      cash = cash + invested;
      invested = 0;
```

```
    end

  end

  returns = cash + invested;

end
```

g) Vi kan ikke bruke negasjonen av is_going_up siden de to hendelsene ikke er komplementære. Dette skyldes at hvis prisendringen er lik null, så kan prisen både være på vei ned og på vei opp.

```
function res = is_going_down( n, i, daily_return )

  res = true;

  for j = i:-1:max(1, i-n+1)
    if (daily_return(j) > 0)
      res = false;
      return;
    end
  end

end
```

```
function returns = contrarian( start_amount, n, closing_price )

  daily_return = daily_returns( closing_price );
  cash = start_amount;
  invested = 0;

  for i = n:size(closing_price, 2);

    invested = invested * closing_price(i) / closing_price(i-1);

    if (is_going_down(n, i, daily_return))
      % Buy
      invested = invested + cash;
      cash = 0;
    else
      % Sell
      cash = cash + invested;
      invested = 0;
    end
  end

  returns = cash + invested;
end
```