Norges teknisk–naturvitenskapelige universitet
Institutt for datateknikk og informasjonsvitenskap

TDT4102 Prosedyre og Objektorientert programmering Vår 2015

Løsningsforslag 0.5
Python/Matlab

Løsningsforslag

1 Kodeforståelse / oversett til Python/Matlab (10%.)

a) Løsning: Oversett følgende kodesnutter til Python.

```python
def isFibonacciNumber(n):
    a = 0
    b = 1
    while b < n:
            temp = b
            b = a + b
            a = temp
    if b == n:
        return True
    else:
        return False
```

Figur 1: Pythonkode

Løsning: Oversett følgende kodesnutter til Matlab.

```
function [ result ] = isFibonacciNumber( n )
    a = 0;
    b = 1;
    while b < n
            temp = b;
            b = a + b;
            a = temp;
    end
    if b == n
        result = true;
    else
        result = false
    end
end
```

Figur 2: Matlab kode

2 Oversett fra Python/Matlab til C++ (90%.)

Oversett følgende kodesnutter til C++ og sjekk at de kompilerer / kjører i ditt IDE.

a) Løsning: Fibonacci rekkerl

```
int fibonacci(int n){
    int a = 0;
    int b = 1;
    cout << "Fibonacci numbers:" << endl;
    for (int x = 1; x < n; x++){
        int temp = b;
        b = a+b;
        a = temp;
        cout << x << " " << b << endl;
    }
    cout << endl;
}
```

Figur 3: C++ kode

b) Løsning: Trekanttall

```
void triangleNumbersBelow(int number){
    int acc = 1;
    int num = 2;
    cout << "Triangle numbers below " << number << ":" << endl;
    while ( acc + num < number){
        acc = acc + num;
        num = num + 1;
        cout << acc << endl;
    }
    cout << endl;
}
```

Figur 4: C++ kode

```
bool isTriangleNumber(int number){
    int acc = 1;
    while (number > 0){
        number = number - acc;
        acc = acc + 1;
    }
    if (number == 0){
        return true;
    }
    else{
        return false;
    }
}
```

Figur 5: C++ kode

c) Løsning: Sum av kvadrerte tall

```
int squareNumberSum(int number){
    int totalSum = 0;
    for (int index = 0; index < number; index++){
        totalSum += index * index;
        cout << index * index << endl;
    }
    cout << totalSum << endl;
    return totalSum;
}
```

Figur 6: C++ kode

d) Løsning: Største av to tall

```
int max(int a, int b){
    if ( a > b ){
        cout << "A is greater than B" << endl;
        return a;
    }
    else{
        cout << "B is greater than or equal A" << endl;
        return b;
    }
}
```

Figur 7: C++ kode

e) Løsning: Primtall 1

```
bool isPrime(int number){
    bool primeness = true;
    for (int divisor = 2; divisor < n; divisor++){
        if (number % divisor == 0){
            primeness = false;
            break;
        }
    }
    return primeness;
}
```

Figur 8: C++ kode

f) Løsning: Primtall 2

```
void naivePrimeNumberSearch(int maxNumber){
    for (int number = 2; number < maxNumber; number ++){
        if (isPrime(number)){
            cout << number << " is a prime " << endl;
        }
    }
}
```

Figur 9: C++ kode

g) Løsning: Største fellesnevner

```
int findGreatestDivisor(int number){
    for (int divisor = number - 1; divisor > 0; divisor--){
        if (number % divisor == 0){
            return divisor;
        }
    }
    return 1;
}
```

Listing 1: C++ kode

h) Løsning: Telling med lister

```
void compareListOfNumbers(int sequence[],int lengthOfSequence){
    int counter[3];
    for (int i = 0; i < 3; i++){
        counter[i] = 0;
    }
    for (int index = 0; index < lengthOfSequence; index++){
        if (sequence[index] < 0){
            counter[0] += 1;
        }
        else if (sequence[index] == 0){
            counter[1] += 1;
        }
        else{
            counter[2] += 1;
        }

    }
    cout << counter[0] << " number were below zero" << endl;
    cout << counter[1] << " number were zero" << endl;
    cout << counter[2] << " numbers were greater than zero" << endl;
}
```

Figur 10: C++ kode