Øving 1: Algoritmer og Datastrukturer

Innhold

Oppgavebeskrivelse

Oppgaven handler om rekursjon. I main.cpp er det to forskjellige rekursive metoder for multiplikasjon. Den sammenligner så hastigheten til de to metodene med std::chrono.

Algoritmen

main.cpp:

```
#include <chrono>
#include <iostream>

using namespace std;

int multMetode1(int n, int x);
int multMetode2(int n, int x);
void tidtaking(int (*funksjon)(int, int), int n, int x);

int main() {
    cout << "Oving 2" << endl;
    int n = 1000;
    int x = 3;

    cout << "n = " << n << ", x = " << x << endl
         << endl;

    cout << "Metode 1: " << endl;
    tidtaking(multMetode1, n, x);
    cout << endl;

    cout << "Metode 2: " << endl;
    tidtaking(multMetode2, n, x);
    cout << endl;

    cout << "Kontroll: " << n * x << endl;

    return 0;
}
```

```
int multMetode1(int n, int x) {
    if (n == 1)
        return x;

    return multMetode1(n - 1, x) + x;
}

int multMetode2(int n, int x) {
    if (n == 1)
        return x;

    if (n & 1)
        return multMetode2(n / 2, x + x) + x;

    return multMetode2(n / 2, x + x);
}

void tidtaking(int (*funksjon)(int, int), int n, int x) {
    int runder = 0;
    int verdi;

    auto start = chrono::high_resolution_clock::now();
    auto finish = chrono::high_resolution_clock::now();

    do {
        verdi = funksjon(n, x);
        finish = chrono::high_resolution_clock::now();
        runder++;
    } while (chrono::duration_cast<chrono::duration<double>>(finish -
start).count() < 1.0);

    cout << "Verdi: " << verdi << endl;
    auto elapsed = chrono::duration_cast<chrono::duration<double,
std::micro>>(finish - start);
    cout << "Tid: " << elapsed.count() / runder << " mikrosekunder per
runde" << endl;
}
```

Tidskompleksitet

Metode 1 har asymptotisk tidskompleksitet på $\Theta(n)$, mens metode 2 har $\Theta(log(n))$. Dette er fordi metode 1 kaller på seg selv n ganger, mens metode 2 halverer n for hver gang den kaller på seg selv slik at etter log(n) kjøringer er n=1 og returneres.

Tidsmålinger

Målingene er gjort med std::chrono over nok runder slik at den totale måletiden tilsvarer 1 sekund og vi får et mer nøyaktig resultat. Kjøretiden er i mikrosekunder.

(n)	Metode 1	Metode 2
100	0.72 ms	0.05 ms
1000	6.65 ms	0.05 ms
10000	67.8 ms	0.05 ms
100000	702.01 ms	0.07 ms

Konklusjon

Etter å ha utført en praktisk tidsmåling med chrono ser vi at målingene for metode 1 stemmer overens med den teoretiske tidskompleksiteten altså $\Theta(n)$, mens målingene for metode 2 gir en så lav verdi at kompleksiteten $\Theta(log(n))$ er vanskelig å se.