C++ Datorlaboration 2

Denna och efterföljande laborationer skrivs och kompileras enklast i Visual Studio.

Uppgift 1: Eratosthenes såll (Eratosthenes Sieve)

Ett enkelt sätt att göra en uppräkning av alla primtal mindre än n är (från Wikipedia):

1. Gör en lista över alla tal från två till n.
2. Stryk ut från listan alla jämna tal som är större än två (4, 6, 8 osv.).
3. Listans nästa tal som inte är utstruket är ett primtal.
4. Stryk ut alla tal, som är både större än det primtalet du hittade i föregående steget och multiplar av det.
5. Upprepa stegen 3 och 4 tills du har nått ett nummer som är större än kvadratroten av n (det största talet i listan).
6. Alla kvarstående tal i listan är primtal.

Uppgiften är att implementera detta. Listan skall vara en c-array av int. Resultatet ska skrivas ut på konsolen.

Strängmanipulation

Uppgifterna här är varianter på samma tema. I en sträng (`input`) så skall en sträng (`before`) ersättas med en teckensekvens (`after`). Det skall göras dels med `std::string`, dels med c-string (`char *`) och till slut på en `istream`. T. ex. så ska uppgift 2 kunna anropas med

```
substitute_str("Hej på dej", "ej", "ig")
```

vilket ska ge resultatet `"Hig på dig"`.
Man ska inte byta ut i det som redan är utbytt dvs.

`substitute_str("Hejjj", "ej", "eje")` ger resultatet `"Hejejj"`.

Uppgift 2: Gör det med std::string

dvs. implementera funktionen:

```
void substitute_str (std::string& iostring,
                     const std::string& before,
                     const std::string& after)
```

Observera att `std::string` är mutable, dvs. det går att ändra på innehållet i en sträng - detta till skillnad från C# där strängar är *immutable*. Parametern `iostring` är både input och output till funktionen ovan.

Tips: undersök `std::string`-metoder som t.ex.: `compare`, `replace` och `find`.

Uppgift 3: Gör det med `char*`

```
char* substitute_cstr(const char* input, const char before, const char* after)
```

resultatsträngen är allokerad på heapen dvs. den som anropar `substitute_cstr` måste deleta strängen.

Observera att `before` är en `char` vilket förenklar uppgiften något.

Observera ett problem i funktionen är att resultatsträngens storlek är okänd. Ett sätt att lösa problemet är att först kolla hur många `before` det finns i `input`, då kan ni räkna ut hur stor resultatsträngen behöver vara.

Uppgift 4: Streams, frivillig uppgift!

Ändra uppgift 2 (`std::string` varianten) till att läsa en stream med tecken och skriva till en stream med tecken.

```
void substitute_strm(std::istream& input,
                     std::istream& output,
                     const std::string& before,
                     const std::string& after)
```