```cpp
/*
 *   Dette er obligg nr. 1 for objekt orientert programmering.
 *
 *   @auther Eskil Refsgaard, NTNU Gjøvik
 *   @file obligg1.cpp
 */

#include <iostream>
#include <iomanip>
#include <string>
#include <vector>
#include "LesData2.h"

using namespace std;

struct Rute
{
    vector<string> stopp;
    int ruteNr,
        totMin;
};

vector<Rute *> gRuter;

const int ANTSTOPP = 11;

const vector<string> gBusstopp = // Navn på alle busstoppene
    {"Skysstasjonen", "Fahlstroms plass", "Sykehuset",
     "Gjovik stadion", "Bergslia", "Overby", "Nybrua",
     "NTNU", "Kallerud", "Hunndalen", "Mustad fabrikker"};

const int gMinutter[ANTSTOPP][ANTSTOPP] =
    {{0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0}, // Skysstasjonen = 0
     {3, 0, 3, 0, 0, 0, 3, 0, 0, 0, 4}, // Fahlstrøms plass = 1
     {0, 3, 0, 1, 0, 0, 0, 0, 0, 0, 0}, // Sykehuset = 2
     {0, 0, 1, 0, 3, 0, 0, 0, 0, 0, 0}, // Gjøvik stadion = 3
     {0, 0, 0, 3, 0, 2, 0, 0, 0, 0, 0}, // Bergslia = 4
     {0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0}, // Øverby = 5
     {0, 3, 0, 0, 0, 0, 0, 2, 0, 0, 2}, // Nybrua = 6
     {0, 0, 0, 0, 0, 0, 2, 0, 0, 4, 0}, // NTNU = 7
     {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0}, // Kallerud = 8
     {0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 2}, // Hunndalen = 9
     {0, 4, 0, 0, 0, 0, 2, 0, 0, 2, 0}};

void skrivMeny();
void nyRute();
void skrivStopp();
void skrivRuter();
void ruteSkrivData(const Rute rute);
void skrivNesteStoppesteder(const int stopp);
bool ruteLesData(Rute &rute);
void slettRute();
void slett();
void slett(const int nr);

/*
 *   Hovedprogrammet
 */

int main()
{

    char svar;
    cout << "Hei og velkommein til programmet mitt <3" << endl;

    skrivMeny();
```

```
67      svar = lesChar("Skriv inn en kommando");
68      while (svar != 'Q')
69      {
70
71          switch (svar)
72          {
73          case 'N':
74              nyRute();       break;
75          case 'A':
76              skrivRuter();   break;
77          case 'B':
78              skrivStopp();   break;
79          case 'S':
80              slettRute();    break;
81          default:
82              skrivMeny();
83          }
84          svar = lesChar("Skriv inn en kommando");
85      }
86      slett();
87  }
88
89  /*
90   *   Funksjon som kaller funksjoner som enten sletter en bestemt
91   *   rute eller alle sammen.
92   *
93   *   @see lesInt() fra "lesData2.h"
94   *   @see slett(const int nr)
95   *   @see slett()
96   */
97
98  void slettRute()
99  {
100     if (gRuter.size() > 0)
101     {
102         cout << "(-1) slett alle, (0) ingen slettes." << endl;
103
104         int temp = lesInt("Hvilken rute skal slettes ", -1, gRuter.size());
105
106         if (temp == -1)
107         {
108             slett(); // Funksksjon som sletter alt
109         }
110         else if (temp > 0)
111         {
112             slett(temp); // Sletter en bestemt
113         }
114         else
115             cout << "Ingen ruter ble slettet" << endl;
116     }
117     else
118         cout << "Det er ingen registrerte ruter" << endl;
119 }
120
121 /*
122  *   Sletter en bestemt rute
123  */
124
125 void slett(const int nr)
126 { // Sletter en enkelt
127
128     delete gRuter[nr - 1];                      // Sletter vector struct variabel
129     gRuter[nr - 1] = gRuter[gRuter.size() - 1]; // Setter bakerste til nr fjernet
130     gRuter.pop_back();                          // Fjerner bakerste
131 }
132
```

```
133 /*
134  *   Sletter alle ruter som har blitt registrert. Den blir
135  *   tilkalt når bruker ønsker å slette rute nr (-1).
136  */
137
138 void slett()
139 {
140     while (!gRuter.empty())
141     {                                       // Så lengde det er noe finnes
142         delete gRuter[gRuter.size() - 1]; // Sletter bakerste
143         gRuter.pop_back();                  // Fjerner bakersteplass
144     }
145     cout << "Alt ble slettet" << endl;
146 }
147
148 /*
149  *   Lager en ny bussrute. Den sjekker om rute består av flere
150  *   enn et stopp. Hvis den gjør det, blir den lagret.
151  *
152  *   @see ruteLesData()
153  *   @see ruteSkrivData()
154  */
155
156 void nyRute()
157 {
158
159     Rute *nyRute;       // Lager ny peker variabel
160     nyRute = new Rute; // Allokerer plass for pekeren
161                         // Hvis totMin < 0, slett allokert plass.
162     if (ruteLesData(*nyRute) == false)
163         delete nyRute;
164     else
165     {
166         gRuter.push_back(nyRute); // Legger ny rute bakerst
167         ruteSkrivData(*nyRute);   // Skriver ut data for nylig laget rute
168     }
169 }
170
171 /*
172  *   Skriver ruten som ble kjørt
173  *   Får tilsendt struct variabel
174  *
175  *   @see skrivStopp()
176  *   @see skrivNesteStoppeSteder()
177  */
178
179 void ruteSkrivData(const Rute rute)
180 {
181
182     cout << "Rute nr. " << rute.ruteNr << endl;
183     cout << "Ruten tok totalt " << rute.totMin << " minutter.\n\n";
184
185     cout << rute.stopp[0]; // Første stoppet
186     for (int i = 0; i < rute.stopp.size() - 1; i++)
187     {                                           // Alle rutene som finnes
188         cout << " --> " << rute.stopp[i + 1]; // Resten av stoppene
189     }
190     cout << "\n\n";
191 }
192 /*
193  *   Funksjon som leser inn data for ny rute. Den bruker
194  *   refferanseoverføring for å hele tiden oppdatere ruten.
195  *
196  *   @param skrivStopp()
197  *   @param skrivNesteStoppeSteder()
198  */
```

```
199
200 bool ruteLesData(Rute &rute)
201 {
202     int stopp, temp = 0;
203     rute.totMin = 0;                              // Nullstiller
204     rute.ruteNr = lesInt("Rute nr: ", 1, 500); // Bestemmer rutenr
205
206     do
207     {
208         if (stopp - 1 == 8)
209             cout << "\nKan ikke dra videre fra "
210                  << gBusstopp[stopp - 1] << "\n\n";
211
212         skrivStopp();
213
214         stopp = lesInt("\nVelg et startsted", 1, 11);
215
216     } while (stopp - 1 == 8); // Kan ikke starte på Kallerud, ingen vei videre
217
218     cout << "\nDu vil starte på stoppet " << gBusstopp[stopp - 1] << endl;
219     (rute.stopp).push_back(gBusstopp[stopp - 1]);
220
221     while (1)
222     { // Kjøres helt til noe returneres.
223
224         skrivNesteStoppesteder(stopp - 1); // Skriver aktuelle stopp
225         temp = stopp;                         // Vite hvor bussen var tidligere
226
227         do
228         { // Kan bare skrive inn lovlige verdier.
229             stopp = lesInt("Hva skal det neste stoppet være: ", 0, 11);
230         } while ((gMinutter[temp - 1][stopp - 1]) == 0 && stopp != 0);
231
232         if (stopp == 0)
233         { // 0 betyr at turen skal stoppes.
234             if (rute.totMin != 0)
235                 return true;
236             else
237                 return false;
238         }
239
240         (rute.stopp).push_back(gBusstopp[stopp - 1]);  // Setter busstoppet
241         rute.totMin += gMinutter[temp - 1][stopp - 1]; // Legger til ant. min
242     }
243 }
244
245 /*
246  *   Skirver neste stopp, avhengig av hvilket stopp en befinner seg på.
247  */
248
249 void skrivNesteStoppesteder(const int stopp)
250 {
251
252     cout << "Neste stopp kan være: \n";
253
254     for (int i = 0; i < ANTSTOPP; i++)
255     {
256         if (gMinutter[stopp][i] != 0)
257         {                                              // Stopp som har vedrdi på linje
258             cout << setw(2) << "nr. " << i + 1    // Hvilket nr
259                  << "\t" << gBusstopp[i] << endl; // Selve stoppet
260         }
261     }
262 }
263
264 /*
```

```
 *   Funksjon som skirver ut alle rutene som er registrert. Bruker
 *   en annen funksjon for å skirve ut selve dataen.
 *
 *   @see ruteSkrivData()
 */

void skrivRuter()
{
    cout << "\n\n**Registrerte bussruter**\n\n";

    for (int i = 0; i < gRuter.size(); i++) // Alle rutene
        ruteSkrivData(*gRuter[i]);          // via peker
}

/*
 *   Skriver alle busstoppen som finnes
 */

void skrivStopp()
{
    for (int i = 0; i < ANTSTOPP; i++)
        cout << setw(2) << i + 1 << ". " << gBusstopp[i] << endl;
}

/*
 *   Skirver menyen for ulik funksjonalitet
 */

void skrivMeny()
{

    cout << "\nDette er menyen;"
         << "\n\tNy rute (N)"
         << "\n\tSlett rute (S)"
         << "\n\tSkriv alle ruter (A)"
         << "\n\tSkriv alle busstopp (B)\n\n";

```