

```

/*****
 * @file    oblig1-R.cpp
 * @brief   Program som gir mulighet for å lage eller slette bussruter
 *
 * @author  Kristupas Kaupas
 * @date    2026-2-10
 *****/

#include <iostream>
#include <iomanip>                //setw()
#include <string>                //string()
#include <vector>
#include "lesdata2.h"           //lesChar(), lesInt()
using namespace std;
const int ANTSTOPP = 11;       ///< Totalt antall ulike busstopp.

struct Rute {
    vector<string> stopp;        // Rutens ulike stoppesteder.
    int ruteNr,                 // Reelt rutenr, f.eks. 42, 165, 718
        totMin;                // Totalt antall minutter å kjøre på ruten
};                               // (fra første til siste stoppested).
vector<Rute*> gRuter;           ///< Pekere til rutene.
const vector<string> gBusstopp = ///< Navn på alle busstopp.
{ "Skysstasjonen", "Fahlstrøms plass", "Sykehuset",
  "Gjøvik stadion", "Bergslia", "Overby", "Nybrua",
  "NTNU", "Kallerud", "Hunndalen", "Mustad fabrikker" };
//const int gMinutter[ANTSTOPP][ANTSTOPP] = ///< Min.mellom stoppesteder.
const vector<vector<int>> gMinutter = // Alternativt (nå lært i vår).
{ { 0, 3, 0, 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, 0 }, // Sykehuset = 2
  { 0, 0, 1, 0, 3, 0, 0, 0, 0, 0, 0, 0 }, // Gjøvik stadion = 3
  { 0, 0, 0, 3, 0, 2, 0, 0, 0, 0, 0, 0 }, // Bergslia = 4
  { 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0 }, // Overby = 5
  { 0, 3, 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, 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 } }; // Mustad fabrikker = 10

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

/**
 * Meny systemet for programmet.
 *
 * @return 0
 */
int main() {
    char input = '\0';
    while (input != 'Q') {
        skrivMeny();

        input = lesChar("Input");
        if (input == 'N') {
            nyRute();
        }
        else if (input == 'S') {
            slettRute();
        }
        else if (input == 'A') {
            cout << "Rutene lagret i systemet er: \n";
            skrivRuter();
        }
        else if (input == 'B') {
            skrivStopp();
        }
    }
    cout << "Program stoppes!!";
    return 0;
}

/**
 * UI - Spør bruker om div info for å lage en ny rute.
 *
 * @see ruteLesData()
 * @see ruteSkrivData();
 */
void nyRute() {
    Rute* nyRute = new Rute;
    if (ruteLesData(*nyRute)) {
        //Lagrer nuRute i gRuter hvis der var mer enn 1 rute i ruteLesData()
        gRuter.push_back(nyRute);
        cout << "Lagret data vises under: \n";
        cout << "Indeks | Rute Nummer | Total reisetid(min) | Alle busstopp \n";
        cout << setw(6) << gRuter.size() << " | ";
        ruteSkrivData(*nyRute);
    }
}

/**
 * UI - Lar bruker sette sammen ruter.
 *
 * @param rute struct
 * @return true/false basert på om det er mer enn 1 stopp som lagres
 * @see skrivStopp()
 * @see skrivNesteStoppesteder()
 */

```

```

bool ruteLesData(Rute& rute) {
    cout << "\n";
    int startStedNr, stedNr;
    bool hentData = true;
    rute.ruteNr = lesInt("Skriv inn rute nummer ", 0, 1000);
    rute.totMin = 0;

    cout << "Velg nr for startsted\n";
    skrivStopp();
    startStedNr = lesInt("Start sted", 1, ANTSTOPP);
    rute.stopp.push_back(gBusstopp[startStedNr - 1]); //lagrer første stoppested

    while(hentData) {
        skrivNesteStoppesteder(startStedNr - 1);
        stedNr = lesInt("Velg neste stoppested. '0' for avslutt ", 0, ANTSTOPP);
        if (stedNr == 0) {
            hentData = false;
        }
        else {
            if (gMinutter[startStedNr - 1][stedNr - 1] != 0) {
                //sjekker om reisetid ikke er 0 altså at den kan nås fra
                // startstopp.
                // Legger til stopp string i struct og plusser på totmin
                rute.stopp.push_back(gBusstopp[stedNr - 1]);
                rute.totMin += gMinutter[startStedNr - 1][stedNr - 1];
            }
            else {
                //reisetid er null - Kan ikke nås fra startstoppet
                cout << "\nStoppet du valgte kan ikke"
                     << "naas fra startstoppet ditt\n";
            }
        }
    }

    if (rute.stopp.size() > 1) {
        return true;
    }
    return false;
}

/**
 * Skriver ut all info om ruten som ble lagret, og dens indeks.
 */
@param rute - Struct
*/
void ruteSkrivData(const Rute rute) {
    cout << setw(11) << rute.ruteNr << " | "
         << setw(19) << rute.totMin << " | ";

    for (int i = 0; i < rute.stopp.size(); i++) {
        cout << rute.stopp[i];
        if (i < rute.stopp.size() - 1) {
            cout << "-- > ";
        }
    }
    cout << "\n";
}

/**
 * UI - Menyvalg tekst i konsollen.
 */
@see main()
*/
void skrivMeny() {
    cout << "\nVelg et valg\n";
    cout << " " << "N = Sett rute\n"
         << " " << "S = Slett rute\n"
         << " " << "A = Skriv ut alle ruter \n"
         << " " << "B = Skriv ut alle busstopp\n"
         << " " << "Q = Avslutt\n\n";
};

/**
 * Skriver lovlige stoppeseder ut fra stopp nr.
 */
@param stopp - int for busstopp indeks
@see ruteLesData()
*/
void skrivNesteStoppesteder(const int stopp) {
    const vector<valgtLinje> = gMinutter[stopp]; //velger en linje fra gMinutter
    // feks {0,0,1,2,3,4..}
    cout << "\nMulige stoppesteder er: \n";
    cout << "-----\n";
    for (int i = 0; i < valgtLinje.size(); i++) {
        if (valgtLinje[i] != 0) {
            //skriver ut alle linjer som nås fra stopp indeks
            // (reisetid ikke null)
            cout << setw(3) << i + 1 << " | " << gBusstopp[i] << "\n";
        }
    }
    cout << "-----\n";
}

/**
 * UI - Skriver alle ruter lagret i systemet.
 */
@see ruteSkrivData()
*/
void skrivRuter() {
    cout << "Indeks | Rute Nummer | Total reisetid(min) | Alle busstopp \n";
    for (int i = 0; i < gRuter.size(); i++) {
        cout << setw(6) << i + 1 << " | ";
        ruteSkrivData(*gRuter[i]);
    }
}

```

```

}

/**
 * UI - Skriver navnet på alle lagrede stoppesteder fra 1 og oppover.
 *
 */
void skrivStopp() {
    cout << "Nr: | Stopnavn \n";
    cout << "-----\n";
    for (int i = 0; i < ANTSTOPP; i++) {
        cout << setw(3)<<i + 1 << " | " << gBusstopp[i] << "\n";
    }
}

/**
 * Sletter alle ruter.
 *
 */
void slett() {
    for (int i = 0; i < gRuter.size(); i++) {
        delete gRuter[i]; //Fjerner data pointers peker til
    }
    gRuter.clear(); //Tømmer hele vektoren
}

/**
 * Sletter en spesifikk rute.
 *
 * @param nr - Int indeks for brukerlagd bussrute
 */
void slett(const int nr) {
    delete gRuter[nr]; //sletter verdien for nr
    swap(gRuter[nr], gRuter.back()); //Bytter plass på bakerst og ønsket element
    gRuter.pop_back(); //Sletter bakerst element
}

/**
 * UI for sletting av ruter.
 *
 * @see skrivRuter()
 * @see slett()
 */
void slettRute() {
    if (gRuter.empty()) {
        cout << "Ingen ruter lagret\n";
    }
    skrivRuter();
    cout << "\n";
    int brukerValg = lesInt(
        "Skriv inn rute index, 0 for ingen, -1 for alle",
        -1, gRuter.size());

    if (brukerValg == -1) {
        slett(); //sletter alle ruter
    }
    else if (brukerValg == 0) {
        cout << "Går tilbake til hovedmeny \n";
    }
    else {
        slett(brukerValg - 1); //slett spesifikk rute
    }
}

```