|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
| // Miestai  #include <fstream>  #include <iomanip>  using namespace std;  const char CDfv[] = "Duomenys.txt";  const char CRfv[] = "Rezultatai.txt";  const int CMax = 300;  const int CPav = 20;  //-----------------------------------------------------------------------  struct Miestas {  string pav, // miesto pavadinimas  valst; // valstybes pavadinimas  int kiek; // miestu skaicius valstybeje  };  //-----------------------------------------------------------------------  void Skaityti(Miestas A[], int & n);  void Spausdinti(Miestas A[], int n, string pav);  void Atrinkti(Miestas A[], int n, Miestas B[], int & m);  int Yra (Miestas A[], int n, string pav);  //-----------------------------------------------------------------------  int main()  {  Miestas A[CMax]; int n; // pradiniu duomenu masyvas  Miestas B[CMax]; int m;  Skaityti(A, n);  Atrinkti(A, n, B, m); // daromas valstybiu sarasas  Spausdinti(B, m, "Valstybiu sarasas:");  return 0;  }  //-----------------------------------------------------------------------  // Skaitomi pradiniai duomenys is failo, nurodyto konstanta CDfv, i masyva A(n)  void Skaityti(Miestas A[], int & n)  {  char eil[CPav + 1]; // papildoma vieta eilutes pabaigos simboliui '\0'  ifstream fd(CDfv);  fd >> n ; // miestu skaicius  fd.ignore(80, '\n'); // visi simboliai failo eiluteje iki jos pabaigos praleidþiami  for (int i = 0; i < n; i++) {  fd.get(eil, CPav); // skaitomas miesto pavadinimas  A[i].pav = eil; // miesto pavadinimas konvertuojamas i string tipo eilutæ  getline(fd, A[i].valst); // skaitomas miesto pavadinimas iki failo eilutes pabaigos  A[i].kiek = 1; // apskaiciuojamas valstybes miestu skaicius  }  fd.close();  }  //-----------------------------------------------------------------------  // Rasomi valstybiu pavadinimai ir nurodoma, kiek toje valstybeje yra miestu  // pav – rezultatu saraso pavadinimas  void Spausdinti(Miestas A[], int n, string pav)  {  ofstream fr (CRfv);  fr << pav << endl;  for (int i = 0; i < n; i++)  fr << setw(15) << A[i].pav << setw(15) << A[i].valst << setw(6) << A[i].kiek << endl;  fr.close();  }  //-----------------------------------------------------------------------  // Formuojamas sarasas, kiek kurioje valstybeje yra miestu  // Masyvas A(n) – pradiniai duomenys, B(m) – rezultatai  void Atrinkti(Miestas A[], int n, Miestas B[], int & m)  {  m = 0; // rezultatu masyvas tuscias  for (int i = 0; i < n; i++) {  int k = Yra(B, m, A[i].valst); // masyve B(m) ieskoma valstybes pavadinimo  if (k >= 0)  B[k].kiek++; // valstybes pavadinimas surastas  else {  B[m] = A[i]; // valstybes pavadinimo nerasta; masyvas B papildomas nauja reiksme  m++;  }  }  }  //-------------------------------------------------------------------  // Masyve A(n) ieskoma, ar pavadinimas pav yra tarp valst  int Yra (Miestas A[], int n, string pav)  {  for (int i = 0; i < n; i++)  if (A[i].valst == pav) return i; // paieska sekminga; graþinamas indeksas,  // kur buvo rastas pav  return -1; // paieska nesekminga; graþinama neigiama reiksme  }  //----------------------------------------------------------------------- | // Miestai  #include <fstream>  #include <iomanip>  using namespace std;  const char CDfv[] = "Duomenys.txt";  const char CRfv[] = "Rezultatai.txt";  const int CMax = 300;  const int CPav = 20;  //-----------------------------------------------------------------------  struct Miestas {  string pav, // miesto pavadinimas  valst; // valstybės pavadinimas  int kiek; // miestų skaičius valstybėje  };  //-----------------------------------------------------------------------  void Skaityti(Miestas A[], int & n);  void Spausdinti(Miestas A[], int n, string pav);  void Atrinkti(Miestas A[], int n, Miestas B[], int & m);  int Yra (Miestas A[], int n, string pav);  // surikiuot valstybių sarasa pagal miestu skaiciu mazejanciai  void Rikiuoti(...);  //-----------------------------------------------------------------------  int main()  {  Miestas A[CMax]; int n; // pradinių duomenų masyvas  Miestas B[CMax]; int m;  Skaityti(A, n);  // patikrinti ar pradiniu duomenu failas netuscias  if ...{  }  else {  Atrinkti(A, n, B, m); // daromas valstybių sąrašas  Spausdinti(B, m, "Valstybių sąrašas:");  Rikiuoti(...);  Spausdinti(... "Surikiuotas sąrašas");  }  return 0;  }  //-----------------------------------------------------------------------  // Skaitomi pradiniai duomenys iš failo, nurodyto konstanta CDfv, i masyva A(n)  void Skaityti(Miestas A[], int & n)  {  char eil[CPav + 1]; // papildoma vieta eilutes pabaigos simboliui '\0'  ifstream fd(CDfv);  fd >> n ; // miestu skaicius  fd.ignore(80, '\n'); // visi simboliai failo eiluteje iki jos pabaigos praleidžiami  for (int i = 0; i < n; i++) {  fd.get(eil, CPav); // skaitomas miesto pavadinimas  A[i].pav = eil; // miesto pavadinimas konvertuojamas i string tipo eilutê  getline(fd, A[i].valst); // skaitomas miesto pavadinimas iki failo eilutes pabaigos  A[i].kiek = 1; // apskaiciuojamas valstybes miestu skaicius  }  fd.close();  }  //-----------------------------------------------------------------------  // Rašomi valstybių pavadinimai ir nurodoma, kiek toje valstybėje yra miestų  // pav – rezultatų sąrašo pavadinimas  void Spausdinti(Miestas A[], int n, string pav)  {  ofstream fr (CRfv);  fr << pav << endl;  for (int i = 0; i < n; i++)  fr << setw(15) << A[i].valst << setw(6) << A[i].kiek << endl;  fr.close();  }  //-----------------------------------------------------------------------  // Formuojamas sarasas, kiek kurioje valstybeje yra miestu  // Masyvas A(n) – pradiniai duomenys, B(m) – rezultatai  void Atrinkti(Miestas A[], int n, Miestas B[], int & m)  {  m = 0; // rezultatu masyvas tušcias  for (int i = 0; i < n; i++){  int k = Yra(B, m, A[i].valst); // masyve B(m) ieškoma valstybes pavadinimo  if (k >= 0)  B[k].kiek++; // valstybes pavadinimas surastas  else {  B[m] = A[i]; // valstybes pavadinimo nerasta; masyvas B papildomas nauja reikšme  m++;  }  }  }  //-------------------------------------------------------------------  // Masyve A(n) ieškoma, ar pavadinimas pav yra tarp valst  int Yra (Miestas A[], int n, string pav)  {  for (int i = 0; i < n; i++)  if (A[i].valst == pav) return i; // paieška sekminga; gr¹žinamas indeksas,  // kur buvo rastas pav  return -1; // paieška nesekminga; gr¹žinama neigiama reikšme  }  //-----------------------------------------------------------------------  void Rikiuoti(...) {  ...  }  } |