My Project

Wygenerowano przez Doxygen 1.9.3

Rozdział 1

Indeks plików

1.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

Rozdział 2

Dokumentacja plików

2.1 Dokumentacja pliku C:/Users/jwiec/Desktop/731ab717-gr11-repo/↩ Projekt/Projekt_Lab/funkcje.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include "Nag.h"
```

Funkcje

- void wpisz1 (List< student > &l, string osoby)
- void wpisz2 (List< oceny > &O, string opis)
- void ListaPrzedmiotow (List< oceny > O, List< string > &przed)
- double sredniaUcznia (string id, string przedmiot, List< oceny > o)
- void srednia (List< student > &s, List< oceny > &O, ofstream &in1)
- void SredniaSzkoly (List< student > &s, List< oceny > &O, double &srednia)
- double SredniaUcznia (List< student > &s, List< oceny > &O, int idx)
- void sredniaOgolna (List< student > &s, List< oceny > &O, ofstream &in)
- void IleRocznikow (List< student > &s, List< string > &Roczniki)
- void WspolnyRok (List< student > &S, ofstream &in)
- void Miasta (List< student > &S, List< string > &miasto)
- void WspolnaMiasto (List< student > &S, ofstream &in)
- void Ulica (List< student > &S, List< string > &ulice)
- void CzyUlica (List< student > &S, string Miasto, List< string > &UliceM)
- void WspolnaUlica (List< student > &S, ofstream &in)
- void Wyswietlacz ()
- void menu (string &osoby, string &opis, string &wyj)

2.1.1 Dokumentacja funkcji

2.1.1.1 CzyUlica()

```
void CzyUlica (
            List< student > & S,
            string Miasto,
            List< string > & UliceM )
```

Definicja w linii 568 pliku funkcje.cpp.

2.1.1.2 IleRocznikow()

```
void IleRocznikow (
            List< student > & s,
            List< string > & Roczniki )
```

Definicja w linii 471 pliku funkcje.cpp.

2.1.1.3 ListaPrzedmiotow()

```
void ListaPrzedmiotow (
            List< oceny > O,
            List< string > & przed )
```

Definicja w linii 360 pliku funkcje.cpp.

2.1.1.4 menu()

```
void menu (
            string & osoby,
            string & opis,
            string & wyj )
```

Definicja w linii 615 pliku funkcje.cpp.

2.1.1.5 Miasta()

```
void Miasta (
            List< student > & S,
            List< string > & miasto )
```

Definicja w linii 521 pliku funkcje.cpp.

2.1.1.6 srednia()

```
void srednia (
            List< student > & s,
            List< oceny > & O,
            ofstream & in1 )
```

Definicja w linii 401 pliku funkcje.cpp.

2.1.1.7 sredniaOgolna()

```
void sredniaOgolna (
            List< student > & s,
            List< oceny > & O,
            ofstream & in )
```

Definicja w linii 456 pliku funkcje.cpp.

2.1.1.8 SredniaSzkoly()

```
void SredniaSzkoly (
            List< student > & s,
            List< oceny > & O,
            double & srednia )
```

Definicja w linii 418 pliku funkcje.cpp.

2.1.1.9 SredniaUcznia()

```
double SredniaUcznia (
            List< student > & s,
            List< oceny > & O,
            int idx )
```

Definicja w linii 437 pliku funkcje.cpp.

2.1.1.10 sredniaUcznia()

```
double sredniaUcznia (
            string id,
            string przedmiot,
            List< oceny > o )
```

Definicja w linii 378 pliku funkcje.cpp.

2.1.1.11 Ulica()

```
void Ulica (
            List< student > & S,
            List< string > & ulice )
```

Definicja w linii 553 pliku funkcje.cpp.

2.1.1.12 wpisz1()

```
void wpisz1 (
            List< student > & l,
            string osoby )
```

Definicja w linii 323 pliku funkcje.cpp.

2.1.1.13 wpisz2()

```
void wpisz2 (
            List< oceny > & O,
            string opis )
```

Definicja w linii 340 pliku funkcje.cpp.

2.1.1.14 WspolnaMiasto()

```
void WspolnaMiasto (
            List< student > & S,
            ofstream & in )
```

Definicja w linii 536 pliku funkcje.cpp.

2.1.1.15 WspolnaUlica()

```
void WspolnaUlica (
            List< student > & S,
            ofstream & in )
```

Definicja w linii 585 pliku funkcje.cpp.

2.1.1.16 WspolnyRok()

```
void WspolnyRok (
            List< student > & S,
            ofstream & in )
```

Definicja w linii 494 pliku funkcje.cpp.

2.1.1.17 Wyswietlacz()

```
void Wyswietlacz ( )
```

Definicja w linii 609 pliku funkcje.cpp.

2.2 funkcje.cpp

Idź do dokumentacji tego pliku.

```
00001
00002 #include<iostream>
00003 #include<fstream>
00004 #include<string>
00005
00006 #include "Nag.h"
00007
00008 using namespace std;
00009
00010 template<typename T>
00011 Element<T>::Element(T Data)
00012 {
00013     Element<T>::data = Data;
00014 }
00015 template<typename T>
00016 void Element<T>::add(T Data)
00017 {
00018     if (next == nullptr)
00019     {
00020         shared_ptr<Element<T» d(new Element(Data));
00021         next = d;
00022     }
00023     else
00024     {
00025         next->add(Data);
00026     }
00027 }
00028 template<typename T>
00029 T& Element<T>::get(unsigned int n)
00030 {
00031     if (n == 0)
00032         return data;
00033     if (next == nullptr)
00034         throw;
00035     else
00036         return next->get(n - 1);
00037 }
00038 template<typename T>
00039 const T& Element<T>::get(unsigned int n) const
00040 {
00041     if (n == 0)
00042         return data;
00043     if (next == nullptr)
00044         throw;
00045     else
00046         return next->get(n - 1);
00047 }
00048 template<typename T>
00049 int Element<T>::size() const
00050 {
00051     if (next == nullptr)
00052         return 1;
```

```
00053     else
00054         return next->size() + 1;
00055 }
00056 template<typename T>
00057 void Element<T>::erase(unsigned int n)
00058 {
00059     if (next == nullptr)
00060         throw;
00061     else if (n == 0)
00062         next = next->next;
00063     else
00064         return next->erase(n - 1);
00065 }
00066 template<typename T>
00067 void Element<T>::swap()
00068 {
00069     std::swap(next, next->next);
00070 }
00071 //-----------------------------------------------------------------------
00072 template<typename T>
00073 List<T>::List(const initializer_list<T> tab)
00074 {
00075     for (T tablica : tab  )
00076     {
00077         List::add(tablica);
00078     }
00079 }
00080 template<typename T>
00081 List<T>::List(const List& datas)
00082 {
00083     int n = datas.size();
00084     for (int i = 0; i < n; i++) {
00085         List::add(datas.get(i));
00086     }
00087 }
00088 template<typename T>
00089 List<T>::List(List&& datas)
00090 {
00091     List::first = datas.first;
00092     datas.first.reset();
00093 }
00094
00095 template<typename T>
00096 void List<T>::add(T Data)
00097 {
00098     if (first == nullptr)
00099     {
00100         shared_ptr<Element<T>» d(new Element<T>(Data));
00101         first = d;
00102     }
00103     else
00104         first->add(Data);
00105 }
00106 template<typename T>
00107 T& List<T>::get(unsigned int n)
00108 {
00109     if (first == nullptr)
00110         throw;
00111     else
00112         return first->get(n);
00113 }
00114 template<typename T>
00115 const T& List<T>::get(unsigned int n) const
00116 {
00117     if (first == nullptr)
00118         throw;
00119     else
00120         return first->get(n);
00121 }
00122 template<typename T>
00123 int List<T>::size() const
00124 {
00125     if (first == nullptr)
00126         return 0;
00127     else
00128         return first->size();
00129 }
00130 template<typename T>
00131 void List<T>::erase(unsigned int n)
00132 {
00133     if (first == nullptr)
00134         throw;
00135     else if (n == 0)
00136         first = first->next;
00137     else
00138         return first->erase(n-1);
00139 }
```

```
00140 template<typename T>
00141 void List<T>::sort()
00142 {
00143     int i, j;
00144     for (i = 0; i < List::size(); i++)
00145     {
00146         for (j = 0; j < List::size() - i - 1; j++)
00147         {
00148
00149             if (first->get(j) > first->next->get(j))
00150             {
00151                 swap(List::get(j), List::get(j + 1));
00152             }
00153         }
00154     }
00155
00156 }
00157
00158 template<typename T>
00159 List<T>& List<T>::operator=(const List& other)
00160 {
00161     int n = other.size();
00162     for (int i = 0; i < n; i++) {
00163         List::add(other.get(i));
00164     }
00165     return *this;
00166 }
00167
00168 template<typename T>
00169 List<T>& List<T>::operator=(List&& other)
00170 {
00171     List::first = other.first;
00172     other.first.reset();
00173     return *this;
00174 }
00175
00176 template<typename T>
00177 void List<T>::serialization(const string& nazwa)
00178 {
00179     ofstream plik(nazwa, ios::binary);
00180     if (!plik) {
00181         std::cout « "blad pliku" « std::endl;
00182         return;
00183     }
00184     auto temp = first;
00185     while (temp) {
00186         T a = temp->data;
00187         plik.write((const char*)&a, sizeof a);
00188         temp = temp->next;
00189     }
00190     plik.close();
00191 }
00192 template<typename T>
00193 void List<T>::deserialization(const string& nazwa) {
00194     ifstream plik(nazwa, ios::binary);
00195     if (!plik) {
00196         std::cout « "Nie mozna otworzyc pliku" « std::endl;
00197         return;
00198     }
00199     T elem;
00200     while (plik.read((char*)(&elem), sizeof(T))) {
00201         this->add(elem);
00202     }
00203     plik.close();
00204 }
00205 //-----------------------------------------------------------------
00206 student::student(string& pesel, string& imie, string& nazwisko, string& data, string& adres, string&
       mieszkanie, string& miasto)
00207 {
00208     student::SetPESEL(pesel);
00209     student::SetImie(imie);
00210     student::SetNazwisko(nazwisko);
00211     student::SetData(data);
00212     student::SetAdres(adres);
00213     student::SetNR_mieszkania(mieszkanie);
00214     student::SetMiasto(miasto);
00215 }
00216 void student::SetPESEL(const string& pesel)
00217 {
00218     student::PESEL = pesel;
00219 }
00220 void student::SetImie(const string& imie)
00221 {
00222     student::imie = imie;
00223 }
00224 void student::SetNazwisko(const string& nazwisko)
00225 {
```

```
00226     student::nazwisko = nazwisko;
00227 }
00228 void student::SetData(const string& data)
00229 {
00230     student::data = data;
00231 }
00232 void student::SetAdres(const string& adres)
00233 {
00234     student::adres = adres;
00235 }
00236 void student::SetNR_mieszkania(const string& mieszkanie)
00237 {
00238     student::nr_mieszkania = mieszkanie;
00239 }
00240 void student::SetMiasto(const string& miasto)
00241 {
00242     student::miasto = miasto;
00243 }
00244 string student::GetPesel()
00245 {
00246     return student::PESEL;
00247 }
00248 string student::GetImie()
00249 {
00250     return student::imie;
00251 }
00252 string student::GetNazwisko()
00253 {
00254     return student::nazwisko;
00255 }
00256 string student::GetData()
00257 {
00258     return student::data;
00259 }
00260 string student::GetAdres()
00261 {
00262     return student::adres;
00263 }
00264 string student::GetNR_mieszkania()
00265 {
00266     return student::nr_mieszkania;
00267 }
00268 string student::GetMiasto()
00269 {
00270     return student::miasto;
00271 }
00272
00273 void student::print(ostream& out)
00274 {
00275     out « GetPesel() « "
       \t"«GetImie()«"\t"«GetNazwisko()«"\t"«GetData()«"\t"«GetAdres()«"\t"«GetNR_mieszkania()«"\t"«GetMiasto();
00276 }
00277 //-------------------------------------------------------
00278 oceny::oceny(string& pesel, string& przedmiot, double& ocena, string& zadanie)
00279 {
00280     oceny::SetPESEL(pesel);
00281     oceny::SetPrzedmiot(przedmiot);
00282     oceny::SetOcena(ocena);
00283     oceny::SetZadanie(zadanie);
00284 }
00285 void oceny::SetPESEL(const string& pesel)
00286 {
00287     oceny::PESEL = pesel;
00288 }
00289 void oceny::SetPrzedmiot(const string& przedmiot)
00290 {
00291     oceny::przedmiot = przedmiot;
00292 }
00293 void oceny::SetOcena(const double& ocena)
00294 {
00295     oceny::ocena = ocena;
00296 }
00297 void oceny::SetZadanie(const string& zadanie)
00298 {
00299     oceny::zadanie = zadanie;
00300 }
00301 string oceny::GetPESEL()
00302 {
00303     return oceny::PESEL;
00304 }
00305 string oceny::GetPrzedmiot()
00306 {
00307     return oceny::przedmiot;
00308 }
00309 double oceny::GetOcena()
00310 {
00311     return oceny::ocena;
```

```
00312 }
00313 string oceny::GetZadanie()
00314 {
00315     return oceny::zadanie;
00316 }
00317
00318 void oceny::print(ostream& out)
00319 {
00320     out « GetPESEL() « "\t" « GetPrzedmiot() « "\t" « GetOcena() « "\t" « GetZadanie();
00321 }
00322
00323 void wpisz1(List<student>& l, string osoby)
00324 {
00325     ifstream out1(osoby);
00326     out1.seekg(0, out1.beg);
00327     string pesel, imie, nazwisko, data, adres, mieszkanie, miasto;
00328     if (out1)
00329     {
00330         while (out1 » pesel » imie » nazwisko » data » adres » mieszkanie » miasto)
00331         {
00332             student s(pesel, imie, nazwisko, data, adres, mieszkanie, miasto);
00333             l.add(s);
00334         }
00335     }
00336     else
00337         cout « "error\n";
00338     out1.close();
00339 }
00340 void wpisz2(List<oceny>& O, string opis)
00341 {
00342     ifstream out(opis);
00343     out.seekg(0, out.beg);
00344     string pesel, przedmiot, zadanie;
00345     double ocena;
00346     if (out)
00347     {
00348         while (out » pesel » przedmiot » ocena » zadanie)
00349         {
00350             oceny d(pesel, przedmiot, ocena, zadanie);
00351             O.add(d);
00352         }
00353     }
00354     else
00355         cout « "error\n";
00356     out.close();
00357
00358 }
00359
00360 void ListaPrzedmiotow(List<oceny> O, List<string>& przed)
00361 {
00362     for (int i = 0; i < O.size(); i++)
00363     {
00364         bool wpis = false;
00365         for (int j = 0; j < przed.size() && !wpis; j++)
00366         {
00367             string temp = O.get(i).GetPrzedmiot();
00368             if (przed.get(j) == temp)
00369                 wpis = true;
00370         }
00371         if (!wpis)
00372         {
00373             przed.add(O.get(i).GetPrzedmiot());
00374         }
00375     }
00376 }
00377
00378 double sredniaUcznia(string id,string przedmiot,List<oceny> o)
00379 {
00380     int ilosc = 0;
00381     double suma = 0;
00382     for (int i = 0; i < o.size(); i++)
00383     {
00384         if (id == o.get(i).GetPESEL() && przedmiot == o.get(i).GetPrzedmiot())
00385         {
00386             suma = suma + o.get(i).GetOcena();
00387             ilosc++;
00388         }
00389     }
00390
00391     if (ilosc == 0)
00392     {
00393         return 1;
00394     }
00395     else
00396     {
00397         double srednia = suma / ilosc;
00398         return srednia;
```

```
00399     }
00400 }
00401 void srednia(List<student>& s,List<oceny>& O, ofstream& in1) //do sredniej z jednego przedmiotu
00402 {
00403
00404     List<string> przed; //przedmioty jakie sa w liscie
00405     ListaPrzedmiotow(O, przed);
00406     for (int i = 0; i < s.size(); i++)
00407     {
00408         in1 « "*" « s.get(i).GetImie() « "  " « s.get(i).GetNazwisko() « "\nSrednia ocena z: \n";
00409         for (int j = 0; j < przed.size(); j++)
00410         {
00411             in1 « przed.get(j) « " - ";
00412             double sr = sredniaUcznia(s.get(i).GetPesel(), przed.get(j), O);
00413             in1 « sr « "\n"; //srednia ocena z konkretnego przedmiotu
00414         }
00415         in1 « "\n";
00416     }
00417 }
00418 void SredniaSzkoly(List<student>& s, List<oceny>& O, double& srednia)
00419 {
00420     int ilosc = 0;
00421     List<string> przed; //przedmioty jakie sa w liscie
00422     ListaPrzedmiotow(O, przed);
00423     for (int i = 0; i < s.size(); i++)
00424     {
00425
00426         for (int j = 0; j < przed.size(); j++){
00427
00428             double sr = sredniaUcznia(s.get(i).GetPesel(), przed.get(j), O);
00429             ilosc++;
00430             srednia = srednia + sr;
00431         }
00432
00433     }
00434     srednia = srednia / ilosc;
00435
00436 }
00437 double SredniaUcznia(List<student>& s, List<oceny>& O,int idx) //do sredniej z jednego przedmiotu
00438 {
00439     int ile = 0;
00440     double SR = 0;
00441     List<string> przed; //przedmioty jakie sa w liscie
00442     ListaPrzedmiotow(O, przed);
00443     for (int i = idx;; )
00444     {
00445
00446         for (int j = 0; j < przed.size(); j++)
00447         {
00448             ile++;
00449             double sr = sredniaUcznia(s.get(i).GetPesel(), przed.get(j), O);
00450             SR = SR + sr;
00451         }
00452         return SR / ile;
00453     }
00454
00455 }
00456 void sredniaOgolna(List<student>& s, List<oceny>& O, ofstream& in)
00457 {
00458     double SredniaSzkolna = 0;
00459     double sredniaucznia = 0;
00460     SredniaSzkoly(s, O, SredniaSzkolna);
00461     in « "Srednia Szkoly : "«SredniaSzkolna«"\n\n";
00462     for (int i = 0; i < s.size(); i++)
00463     {
00464         in « s.get(i).GetPesel() « "\t" « s.get(i).GetImie() « "  " « s.get(i).GetNazwisko() «
      "\nSrednia ocena : ";
00465         sredniaucznia = SredniaUcznia(s, O, i);
00466         in « sredniaucznia«"\n\n";
00467     }
00468     in « endl;
00469
00470 }
00471 void IleRocznikow(List<student>& s, List<string>& Roczniki)
00472 {
00473     for (int i = 0; i < s.size(); i++)
00474     {
00475
00476         string temp1 = s.get(i).GetData();
00477         string temp2;
00478         for (int k = temp1.size() - 4; k < temp1.size(); k++) //zapisuje roczniki z daty
00479         {
00480             temp2.push_back(temp1[k]);
00481
00482         }
00483         //cout « temp2;
00484         bool wpis = false;
```

```
00485         for (int j = 0; j< Roczniki.size() &&!wpis; j++)
00486         {
00487                 if (Roczniki.get(j) == temp2)
00488                     wpis = true;
00489         }
00490         if (!wpis)
00491             Roczniki.add(temp2);
00492     }
00493 }
00494 void WspolnyRok(List<student>& S, ofstream& in)
00495 {
00496     List<string> Roczniki;
00497     IleRocznikow(S, Roczniki);
00498     Roczniki.sort();
00499     in « "*Uczniowie z rocznika: \n";
00500     for (int j = 0; j < Roczniki.size(); j++)
00501     {
00502         in  «"-" « Roczniki.get(j) « ":\n";
00503         for (int i = 0; i < S.size(); i++)
00504         {
00505             string temp1 = S.get(i).GetData();
00506             string temp2;
00507             for (int k = temp1.size() - 4; k < temp1.size(); k++) //zapisuje roczniki z daty
00508             {
00509                 temp2.push_back(temp1[k]);
00510
00511             }
00512             if (Roczniki.get(j) == temp2) {
00513                 in « S.get(i).GetImie()«"  "«S.get(i).GetNazwisko()«"  "«S.get(i).GetData()«" \n";
00514             }
00515
00516
00517         }
00518             in « endl;
00519     }
00520 }
00521 void Miasta(List<student>& S, List<string>& miasto)
00522 {
00523     for(int i = 0; i < S.size(); i++)
00524     {
00525         bool wpis = false;
00526         for (int j = 0; j < miasto.size() && !wpis; j++)
00527         {
00528             string temp = S.get(i).GetMiasto();
00529             if (miasto.get(j) == temp)
00530                 wpis = true;
00531         }
00532         if (!wpis)
00533             miasto.add(S.get(i).GetMiasto());
00534     }
00535 }
00536 void WspolnaMiasto(List<student>& S, ofstream& in)
00537 {
00538     List<string> miasto;
00539     Miasta(S, miasto);
00540     in « "*Uczniowie mieszkajacy w jednym miescie: \n";
00541     for (int i = 0; i < miasto.size(); i++)
00542     {
00543         in «"-" « miasto.get(i) « " : \n";
00544         for (int j = 0; j < S.size(); j++)
00545         {
00546             string temp = S.get(j).GetMiasto();
00547             if (miasto.get(i) == temp)
00548                 in « S.get(j).GetImie() « "  " « S.get(j).GetNazwisko() « "\n";
00549         }
00550         in « endl;
00551     }
00552 }
00553 void Ulica(List<student>& S, List<string>& ulice)
00554 {
00555     for (int i = 0; i < S.size(); i++)
00556     {
00557         bool wpis = false;
00558         for (int j = 0; j < ulice.size() && !wpis; j++)
00559         {
00560             string temp = S.get(i).GetAdres();
00561             if (ulice.get(j) == temp)
00562                 wpis = true;
00563         }
00564         if (!wpis)
00565             ulice.add(S.get(i).GetAdres());
00566     }
00567 }
00568 void CzyUlica(List<student>& S, string Miasto,List<string>& UliceM)
00569 {
00570     for (int i = 0; i < S.size(); i++)
00571     {
```

```
00572         if (S.get(i).GetMiasto()==Miasto) {
00573             bool wpis = false;
00574             for (int j = 0; j < UliceM.size(); j++)
00575             {
00576                 string temp = S.get(i).GetAdres();
00577                 if (UliceM.get(j) == temp)
00578                     wpis = true;
00579             }
00580             if (!wpis)
00581                 UliceM.add(S.get(i).GetAdres());
00582         }
00583     }
00584 }
00585 void WspolnaUlica(List<student>& S, ofstream& in)
00586 {
00587     List<string> miasto;
00588     Miasta(S, miasto);
00589     in « "Uczniowie mieszkajacy w jednym miescie i na jednej ulicy: \n\n";
00590     for (int i = 0; i < miasto.size(); i++)
00591     {
00592         in « "*"«miasto.get(i) « ": \n";
00593         List<string> Ulicem;
00594         CzyUlica(S, miasto.get(i), Ulicem);
00595         for (int j = 0; j < Ulicem.size(); j++)
00596         {
00597             in «"-"« Ulicem.get(j) « ": \n";
00598             for (int k = 0; k < S.size(); k++)
00599             {
00600                 if(S.get(k).GetAdres()==Ulicem.get(j) && S.get(k).GetMiasto()==miasto.get(i))
00601                     in«  S.get(k).GetImie() « "  " « S.get(k).GetNazwisko() « "\n";
00602             }
00603             in « endl;
00604         }
00605         in « endl;
00606 
00607     }
00608 }
00609 void Wyswietlacz()
00610 {
00611     cout « "Dziennik Szkolny\n1. Srednia uczniow z konkretnych przedmiotow\n2. Srednia szkoly i
       srednia ocen uczniow z calej ich pracy\n";
00612     cout « "3. Wypisz uczniow po roku urodzenia\n4. Wypisz uczniow mieszkajacych we wspolnych
       miastach\n5. Wypisz uczniow ktorzy mieszkaja na jednej ulicy\n";
00613     cout«"Wybierz numer pliku jaki chcesz wygenerowac : ";
00614 }
00615 void menu(string& osoby, string& opis, string& wyj)
00616 {
00617     List<student> S;
00618     List<oceny> O;
00619     wpisz1(S, osoby);
00620     wpisz2(O, opis);
00621     ofstream in(wyj);
00622     string numer;
00623     Wyswietlacz();
00624     cin » numer;
00625     if (numer == "1")
00626         srednia(S, O, in);
00627     else if (numer == "2")
00628         sredniaOgolna(S, O, in);
00629     else if (numer == "3")
00630         WspolnyRok(S, in);
00631     else if (numer == "4")
00632         WspolnaMiasto(S, in);
00633     else if (numer == "5")
00634         WspolnaUlica(S, in);
00635     else
00636         cout « "\nNie ma opcji generowania pliku z takim numerem";
00637 
00638 }
```

2.3 Dokumentacja pliku C:/Users/jwiec/Desktop/731ab717-gr11-repo/↩ Projekt/Projekt_Lab/main.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include "Nagłówek.h"
```

Funkcje

- int main (int argc, const char ∗argv[ ])

2.3.1 Dokumentacja funkcji

2.3.1.1 main()

```
int main (
              int argc,
              const char * argv[] )
```

Funkcja main, punkt wyjścia do wykonywania programu

Parametry

argc	liczba parametrow przy uruchomieniu programu
argv	tablica wskaznikow na lancuchy uzytych przy uruchomienieu programu

Zwraca

jeśli program wykonał się prawidłowo zwracane jest 0

Definicja w linii 15 pliku main.cpp.

2.4 main.cpp

Idź do dokumentacji tego pliku.

```
00001 #include<iostream>
00002 #include<fstream>
00003 #include<string>
00004 #include <vector>
00005
00006 #include "Nagłówek.h"
00007
00015 int main(int argc, const char* argv[])
00016 {
00017     string osoby, opis, wyjsciowy;
00018     string napis;
00019     if (argc == 7)
00020     {
00021         for (int i = 0; i < argc - 1; i++)
00022         {
00023             napis = argv[i];
00024             if(napis == "-inper")
00025                 osoby= argv[i + 1];
00026             else if(napis== "-insub")
00027                 opis = argv[i + 1];
00028             else if(napis == "-out")
00029                 wyjsciowy= argv[i + 1];
00030         }
00031     }
00032     else
00033     {
00034         cout « "Potrzebne parametry do poprawnego wlaczenia programu\n-inper \tplik z osobami \n-insub
      \tplik z ocenami \n-out \tplik wyjsciowy\n"« endl;
00035         return 0;
00036     }
00037     menu(osoby, opis, wyjsciowy);
00038 }
```

2.5 Dokumentacja pliku C:/Users/jwiec/Desktop/731ab717-gr11-repo/↩ Projekt/Projekt_Lab/Nagłówek.h

2.6 Nagłówek.h

Idź do dokumentacji tego pliku.

Indeks