Programowanie Obiektowe – Projekt 7 C++

Marcin Krueger 145244

Opis plików:

- project.cpp – plik główny projektu, z którego wywoływana jest instancja klasy Interface
- Interface.h, Interface.cpp – pliki, które odpowiadają za rozpoznawanie komend i wywołanie odpowiednich funkcji. Skupiają wszystkie klasy i wykonują na nich właściwe operacje.
  Pola:
  - std::string cmds[11] – przechowuje wszystkie możliwe operacje
  - std::string node[11] – przechowuje nazwy węzłów w drzewie klas
  - std::string node_name[11] – przechowuje pełne nazwy klas (po polsku)
  - std::string curr_pos – przechowuje nazwę węzła, w którym aktualnie się znajdujemy
  - bool is_leaf – przechowuje informacje czy węzeł jest liściem
  - bool end_app – przechowuje informacje czy program ma się zakończyć

  Metody:
  - Interface() - konstruktor klasy Interface
  - ~Interface() - destruktor klasy Interface
  - void help() - odpowiada za komendę HELP, pokazuje na ekranie komunikat tekstowy, który zawiera wszystkie możliwe komendy, znaczniki <> oznaczają że powinno się wpisać odpowiednią wartość, natomiast znaczniki <<>> dotyczą jedynie klasy Study (Studium języków obcych)
  - void exit() - zamyka program
  - void getCommand() - pobiera ze strumienia wejściowego dane wpisane przez użytkownika i w odpowiedni sposób je interpretuje
  - void showDirectory() - odpowiada za komendę DIR
  - void saveToFile(std::string) – odpowiada za komendę SAVE
  - void readFromFile(std::string) – odpowiada za komendę READ
  - void tree() - odpowiada za komendę TREE
- University.h, University.cpp – pliki odpowiadające za korzeń – klasę University (Uczelnia_wyższa).
  Pola:
  - std::string collage_name – przechowuje nazwę uczelni
  - std::string address – przechowuje adres uczelni
  - std::string class_name – przechowuje nazwę klasy

  Metody:
  - University() - konstruktor klasy University
  - ~University() - destruktor klasy University
  - void showAtributes() - funkcja czysto wirtualna wykorzystywana w liściach do wyświetlania informacji o obiekcie
  - std::string getCollageName() - zwraca nazwę uczelni
  - std::string getCollageAddress() - zwraca adres uczelni
  - std::string getClassName() - zwraca nazwę klasy

- Deanary_el.h, Deanary_el.cpp – pliki odpowiadające za klasę Deanary_el (Dziekanat_el).
  Pola:
  - int field_amount – przechowuje ilość podklas
  - std::string depart_name – przechowuje nazwę wydziału
  - std::string office_address – przechowuje adres dziekanatu
  - std::string class_name – przechowuje nazwę klasy

Metody:
  - Deanary_el() - konstruktor klasy Deanary_el
  - ~Deanary_el() - destruktor klasy Deanary_el
  - std::string getClassName() - zwraca nazwę klasy
  - std::string getDepartName() - zwraca nazwę wydziału
  - std::string getDepartAddress() - zwraca adres wydziału

- Deanary_bm.h, Deanary_bm.cpp – pliki odpowiadające za klasę Deanary_bm (Dziekanat_bm).
  Pola:
  - int field_amount – przechowuje ilość podklas
  - std::string depart_name – przechowuje nazwę wydziału
  - std::string office_address – przechowuje adres dziekanatu
  - std::string class_name – przechowuje nazwę klasy

  Metody:
  - Deanary_bm() - konstruktor klasy Deanary_bm
  - ~Deanary_bm() - destruktor klasy Deanary_bm
  - std::string getClassName() - zwraca nazwę klasy
  - std::string getDepartName() - zwraca nazwę wydziału
  - std::string getDepartAddress() - zwraca adres wydziału

- Deanary_bl.h, Deanary_bl.cpp – pliki odpowiadające za klasę Deanary_bl (Dziekanat_bl).
  Pola:
  - int field_amount – przechowuje ilość podklas
  - std::string depart_name – przechowuje nazwę wydziału
  - std::string office_address – przechowuje adres dziekanatu
  - std::string class_name – przechowuje nazwę klasy

  Metody:
  - Deanary_bl() - konstruktor klasy Deanary_bl
  - ~Deanary_bl() - destruktor klasy Deanary_bl
  - std::string getClassName() - zwraca nazwę klasy
  - std::string getDepartName() - zwraca nazwę wydziału
  - std::string getDepartAddress() - zwraca adres wydziału

- Institute_XXX.h, Institute_XXX.cpp – pliki odpowiadające za klasy: Institute_inf (Inst_inf), Institute_el (Inst_elek), Institute_mch (Inst_mech), Institute_phs (Inst_fiz), Institute_rc (Inst_drog), Institute_rt (Inst_kolej).
  Pola:
  - int worker_amount – przechowuje ilość pracowników
  - std::string name – przechowuje imię i nazwisko pracownika
  - std::string title – przechowuje tytuł pracownika
  - std::string email – przechowuje adres e-mail pracownika
  - std::string phone – przechowuje numer telefonu pracownika
  - int thesis_amount – przechowuje ilość napisanych prac naukowych
  - std::string class_name – przechowuje nazwę klasy

Metody:
  - Institute_XXX(std::string, std::string, std::string, std::string, int) – konstruktor klasy Institute_XXX, dodaje obiekt do wektora klasy LeafArray
  - ~Institute_XXX() - destruktor klasy Institute_XXX
  - void showAtributes() - wyświetla atrybuty obiektu
  - int getWorkerAmount() - zwraca ilość pracowników w instytucie
  - std::string getClassName() - zwraca nazwę klasy
  - void modifyObject(std::string, std::string, std::string, std::string, int) – modyfikuje obiekt
  - void saveToFile(std::fstream&) - zapisuje odpowiednie dane do pliku wykorzystując przeciążenie operatora <<
  - std::string getName() - zwraca imię i nazwisko pracownika
  - std::string getTitle() - zwraca tytuł pracownika
  - std::string getEmail() - zwraca e-mail pracownika
  - std::string getPhone() - zwraca numer telefonu pracownika
  - int getThesisAmount() - zwraca liczbę napisanych artykułów naukowych

- Study.h, Study.cpp – pliki odpowiadające za klasę Study (Studium).
  Pola:
  - int worker_amount – przechowuje ilość pracowników
  - std::string name – przechowuje imię i nazwisko pracownika
  - std::string title – przechowuje tytuł pracownika
  - std::string email – przechowuje adres e-mail pracownika
  - std::string phone – przechowuje numer telefonu pracownika
  - std::string language – przechowuje uczony język
  - int thesis_amount – przechowuje ilość napisanych prac naukowych
  - std::string class_name – przechowuje nazwę klasy

  Metody:
  - Study(std::string, std::string, std::string, std::string, int, std::string) – konstruktor klasy Study, dodaje obiekt do wektora klasy LeafArray
  - ~Study() - destruktor klasy Study
  - void showAtributes() - wyświetla atrybuty obiektu
  - int getWorkerAmount() - zwraca ilość pracowników w instytucie
  - std::string getClassName() - zwraca nazwę klasy
  - void modifyObject(std::string, std::string, std::string, std::string, int) – modyfikuje obiekt
  - void saveToFile(std::fstream&) - zapisuje odpowiednie dane do pliku wykorzystując przeciążenie operatora <<
  - std::string getName() - zwraca imię i nazwisko pracownika
  - std::string getTitle() - zwraca tytuł pracownika
  - std::string getEmail() - zwraca e-mail pracownika
  - std::string getPhone() - zwraca numer telefonu pracownika
  - std::string getLanguage() - zwraca uczony język
  - int getThesisAmount() - zwraca liczbę napisanych artykułów naukowych

- LeafArray.h – klasa generyczna, zawiera wektor wskaźników na obiekty odpowiednich klas.
  Pola:
  - int array_size – przechowuje ilość elementów w wektorze
  - std::vector<T*> l_array – wektor przechowujący wskaźniki na obiekty

Metody:

- void addElement(T*) - dodaje element do wektora
- void showElement(std::string) – wyświetla informacje dotyczące obiektu (wykorzystane w komendzie SHOW)
- void showObjects() - wyświetla wszystkie elementy wektora
- void removeElement(std::string) – usuwa element z wektora i z pamięci (wykorzystane w komendzie DO)
- void modifyElement(std::string, std::string, std::string, std::string, std::string, int) – modyfikuje obiekt Inst_XXX o nazwie podanej w pierwszym argumencie (wykorzystane w komendzie MDO)
- void modifyStudy(std::string, std::string, std::string, std::string, std::string, std::string, int) – modyfikuje obiekt Study o nazwie podanej w pierwszym argumencie (wykorzystane w komendzie MDO)
- void saveAll (std::fstream&) - zapis wszystkich obiektów w wektorze do pliku (wykorzystane w komendzie SAVE)

W projekcie wykorzystano także przeciążenie operatorów zdefiniowanie w każdym liściu. Obciążony został operator <<, który ma teraz również za zadanie zapis w odpowiednim formacie danych obiektu do pliku.