Warszawa, 25.01.2017 r.

Politechnika Warszawska
Wydział Elektroniki i Technik Informacyjnych

Oprogramowanie Systemów Medycznych (OSM)

Projekt #1: Tworzenie interfejsu graficznego aplikacji

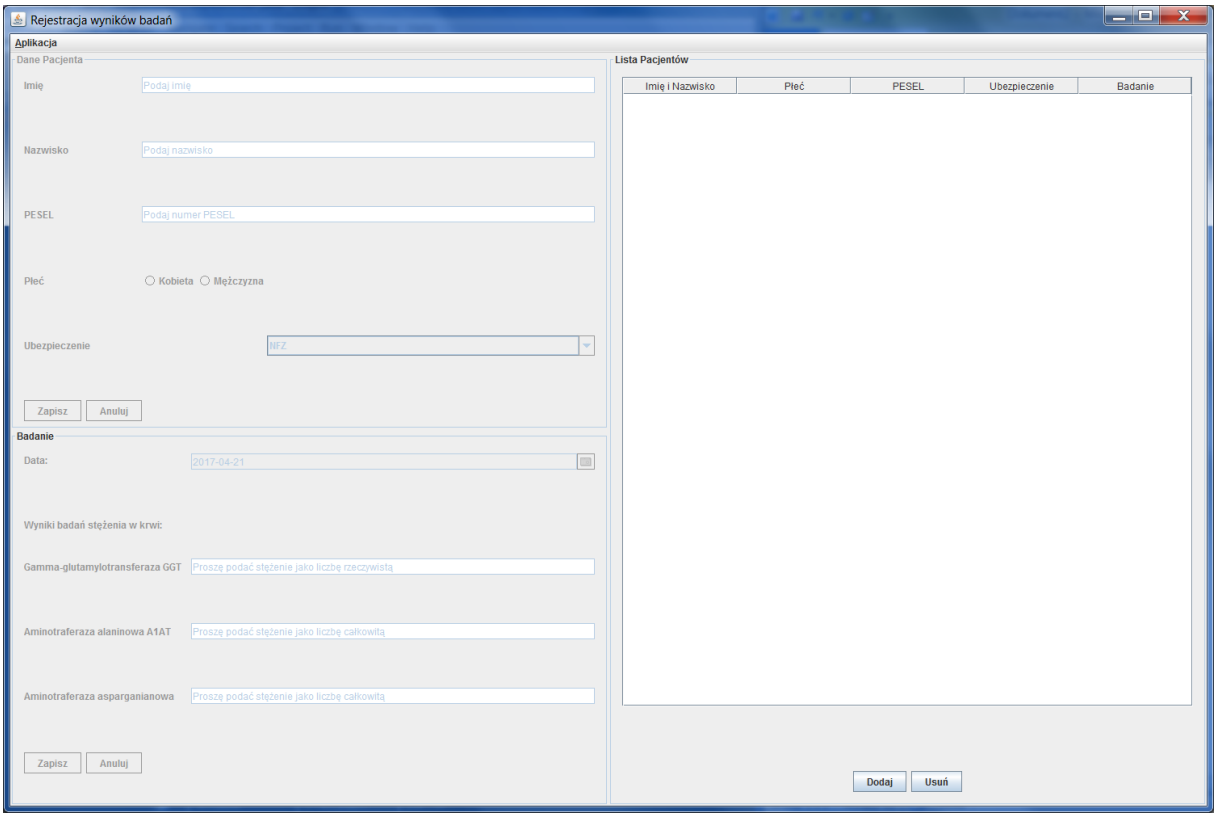
Zadanie #**16**: Biomarkery nadużywania alkoholu: stężenie gamma-glutamylo-transferazy GGT (liczba rzeczywista) oraz poziomy aminotransferaz alaninowej AlAT i asparaginianowej AspAT (liczby całkowite).

Prowadzący projekt: dr inż. Tymon Rubel

Wykonawcy: Weronika Borucka, Adam Jackowski

1. Opis działania programu:

Po uruchomieniu, program wyświetla się w następujący sposób:



Aktywny jest tylko panel po lewej stronie – panel Lista. Po kliknięciu przycisku „Dodaj”, aktywuje się panel odpowiedzialny za dane pacjenta – panel Pacjenta. Użytkownik może teraz wpisywać dane: Imię, Nazwisko oraz numer Pesel. Dodatkowo, może on wybrać płeć, klikając na odpowiednią opcję (domyślna opcja – kobieta) oraz rodzaj ubezpieczenia z listy rozwijanej (prywatne, NFZ, brak, domyślnie ustawiona jest opcja „prywatne”). Między polami można poruszać się, poza klikaniem myszką na odpowiednie pola, za pomocą klawisza „Tab”. Każde pole jest czyszczone, gdy zostanie ono kliknięte bądź wybrane klawiszem „Tab”.

Po kliknięciu przycisku „Anuluj”, wszystkie pola są kasowane, a wartości „płeć” i „ubezpieczenie”{ pozostają zaznaczone tak, jak były ustawione przed kliknięciem przycisku. Panel Pacjenta pozostaje aktywny. Następnie, po kliknięciu przycisku „Zapisz” dane pacjenta są sprawdzane. Gdy wprowadzone dane nie odpowiadają założeniom, pojawiają się odpowiednie okna dialogowe, mówiące o rodzaju błędu (wypisane poniżej w punkcie „Rodzaje obsługiwanych błędów”). Natomiast, gdy wprowadzone dane są poprawne, pacjent jest zapisywany, dodawany jest on jednocześnie na liście obok (w panelu Lista), panel Pacjenta jest wyłączany, natomiast włączony zostaje panel Badania. Ostatnio dodany pacjent jest automatycznie zaznaczany na panelu Listy, dlatego też można do niego od razu dodać odpowiednie dane badania.

Użytkownik wprowadza dane badania, również możliwe jest poruszanie się klawiszem „Tab” po panelu. Dostępne pola to stężenie GGT, A1AT oraz AspAT, dodatkowo należy również wybrać datę. Datą domyślnie ustawioną jest dzień, w którym program został uruchomiony.

Po kliknięciu przycisku „Anuluj”, panel Pacjenta jest wyłączany, aktywny pozostaje tylko panel Listy, z zaznaczonym ostatnio dodanym pacjentem. Po kliknięciu przycisku „Zapisz”, następuje sprawdzenie danych dotyczących badania. Jeśli dane są niepoprawne, pojawiają się odpowiednie okna

dialogowe, mówiące o rodzaju błędu. Dodatkowo, źle wprowadzone dane zaznaczane są kolorem czerwonym. Kolor zmienia się na czarny, jeśli np. po niepoprawnym wprowadzeniu dwóch wartości tylko jedna z nich zostanie poprawiona, a użytkownik naciśnie przycisk „Zapisz".

Aby edytować danego pacjenta z listy, trzeba na niego kliknąć. W przypadku, gdy zachodzi potrzeba edytowania ostatnio dodanego pacjenta, trzeba kliknąć w nieużywaną przestrzeń w programie, po czym kliknąć na pacjenta jeszcze raz. Po wyborze pacjenta z listy, jego dane pojawiają się na panelu Pacjenta oraz w panelu Badania, o ile badanie dla konkretnego pacjenta zostało wcześniej zapisane.

Po kliknięciu na przycisk „Usuń", na panelu Listy, z listy usuwany jest aktualnie zaznaczony na panelu pacjent.

Aby uruchomić program, należy w środowisku Eclipse otworzyć projekt pod nazwą ProjektOSM1, dołączyć do niego odpowiednią bibliotekę „jcalendar-1.4.jar" i uruchomić go za pomocą komendy „Run".

2. Rodzaje obsługiwanych błędów

Błędy mogące wystąpić podczas zapisywania pacjenta:

- wprowadzenie niepoprawnego imienia (ze znakiem nie będącym literą)
- wprowadzenie niepoprawnego nazwiska (ze znakiem nie będącym literą)
- wprowadzenie niepoprawnego formatu numeru PESEL (występowanie znaków nie będących cyfrą)
- wprowadzenie niepoprawnej długości numeru PESEL (krótszy lub dłuższy niż 11 cyfr)

Błędy mogące wystąpić podczas zapisywania badania:

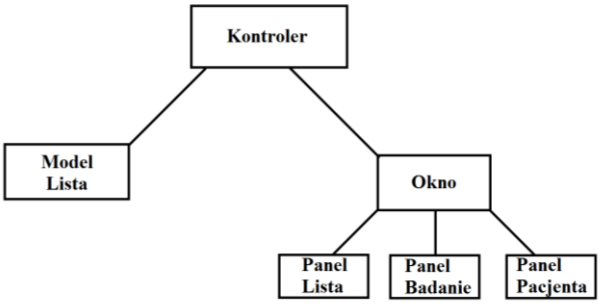
- wprowadzenie niepoprawnej formy stężenia (dla GGT – musi być liczbą rzeczywistą, dla A1AT – liczbą całkowitą, dla AspAT – liczbą całkowitą)
- wprowadzenie niepoprawnej wartości stężenia (dla każdego parametru poprawne stężenie ustawione jest w zakresie stężenie>0 && stężenie<100)
- wprowadzenie niepoprawnego formatu daty

3. Schemat programu

Program jest wykonany według założeń MVC – *Model Viev Controller*

- Widok – zawiera cały interfejs GUI;
  - Okno *(główne okno, spinające 3 poniższe panele)*
  - PanelPacjenta
  - PanelBadania
  - PanelLista
- Model – definicja Pacjenta i Badania; przechowuje pacjentów (ArrayList lista)
  - ModelLista
  - Pacjent
  - Badanie
- Kontroler – odpowiada za kontrolę działania użytkownika
  - Kontroler

Program uruchamiany jest za pomocą pliku „PokazOkno", w którym znajduje się funkcja main.



4. Opis poszczególnych klas i plików

a) Badanie

Jest to klasa odpowiadająca za tworzenie badania, które następnie w toku budowania programu przypisywane jest do obiektu typu Pacjent.

Pola:

- GGT – stężenie gamma-glutamylotransferazy GGT
- A1AT – stężenie aminotraferazy alaninowej A1AT
- AspAT – stężenie aminotraferazy asparganianowej
- data – data typu Date

Metody:

- gettery i settery – automatycznie wygenerowane metody, pozwalające na ustawienie wartości prywatnych pól dla pacjenta oraz ich pobieranie
- hashCode oraz equals - automatycznie wygenerowane metody hashcode i equals

b) Pacjent

Jest to klasa odpowiadająca za tworzenie obiektów typu pacjent.

Pola:

- imie – pole prywatne, rodzaj: string
- nazwisko – pole prywatne, rodzaj: string
- pesel – pole prywatne, rodzaj: string
- badanie – pole prywatne, rodzaj: Badanie - klasa stworzona na potrzeby programu, opisana w poprzednim punkcie, pierwotnie pole to jest puste, ponieważ najpierw tworzony jest pacjent, a następnie dopiero badanie, które jest do niego przypisywane
- plec – pole prywatne, rodzaj: EnumPlec - stworzony na potrzeby programu, zawierający 2 wartości: KOBIETA oraz MEZCZYZNA.
- ubezpieczenie – pole prywatne, rodzaj: EnumUbezpieczenie - stworzony na potrzeby programu, zawierający 3 wartości: NFZ, PRYWATNE, BRAK.

Metody:

- gettery i settery – automatycznie wygenerowane metody, pozwalające na ustawienie wartości prywatnych pól dla pacjenta oraz ich pobieranie

- isBadanie - sprawdza czy badanie istnieje
- hashCode oraz equals - automatycznie wygenerowane metody hashcode i equals, equals – pozwala porównać wszystkie pola danych dwóch obiektów (pacjentów w tym przypadku), nie tylko ich referencje

c) Panel Pacjenta

Panel Pacjenta jest fragmentem interfejsu użytkownika. Umożliwia użytkownikowi wprowadzenie podstawowych danych pacjenta, bez wyników badań. Jest jednym z 3 paneli okna głównego (razem z Panelem Badania i Panelem Listy).

Layout:

Zastosowany został GroupLayout. Panel został podzielony na 2 główne kolumny. W każdej kolumnie jest równa ilość wierszy. Ostatni wiersz zawiera więcej elementów niż wcześniejsze wiersze. Jest to spowodowane podziałem jeden komórki na 2 oddzielne części. W częściach tych zostały umieszczone przyciski Zapisz i Anuluj.

Pola:

- JLabel – obiekt będący polem z wyświetlonym tekstem. W panelu obiekt klasy JLabel informuje jakie informacje należy wpisać lub podać w polu po prawej stronie.
- JTextField – obiekt będący polem w które wpisujemy dane pacjenta. Podczas startu programu w polu jest podpowiedź, jakie wartości należy podać. Przy aktywacji danego pola podpowiedź znika i kursor ustawia się z lewej strony pola tekstowego.
- JComboBox – obiekt będący rozwijaną listą wyboru. Po naciśnięciu na obiekt tej klasy, rozwinięta zostaje lista, po naciśnięciu na konkretny element zostaje on wybrany. Elementu dodajemy poprzez formułę additem().
- ButtonGroup – obiekt będący połączeniem kilku przycisków. Przyciski są wyświetlane w jednej linii.
- JRadioButton – obiekt będący polem z wyświetlonym tekstem z polem do zaznaczania. Po naciśnięciu na kropkę zostaje wybrany obiekt reprezentujący to, co napisane jest w polu do wyświetlania tekstu.
- JButton – obiekt będący prostokątnym przyciskiem. W konstruktorze wpisujemy tekst, wyświetlany na przycisku. Jego naciśnięcie wysyła informację do kontrolera, gdzie wykonuje się stosowna operacja opisana w klasie kontroler.

Metody:

- setKontroler(ActionListener, FocusListener) – metoda ustawia słuchacza zdarzeń dla obiektów TextField i JButton. Po naciśnięciu na te pola do kontrolera zostaje wysyłana stosowna informacja i wykonuje on działanie. ActionListener obsługuje przyciski, FocusListener obsługuje pola tekstowe.
- Gettery i Settery – pobierają wartość z pola tekstowego lub ustawiają wartość pola tekstowego. Pobierają lub tworzą obiekt klasy String, nie obiekt JTextField.
- UstawPacjenta(Pacjent) – metoda pobiera obiekt klasy Pacjent i uzupełnia pola klasy PanelPacjenta wartościami obiektu Pacjent.
- WyczyscPacjenta – metoda czyści wszystkie pola klasy PanelPacjenta i ustawia je na wartość domyślną.
- CzySlowo – metoda pobiera łańcuch znaków i sprawdza czy każdy znak jest literą. Zwraca wartość logiczną true lub false.

d) Panel Badania

Panel Badania jest fragmentem interfejsu użytkownika. Umożliwia użytkownikowi wprowadzenie wyników badań w odpowiedniej formie oraz daty badania. Jest jednym z 3 paneli okna głównego (razem z Panelem Pacjenta i Panelem Listy).

Layout:

Zastosowany został GroupLayout. Panel został podzielony na 2 główne kolumny. W każdej kolumnie jest równa ilość wierszy. Ostatni wiersz zawiera więcej elementów niż wcześniejsze wiersze. Jest to spowodowane podziałem jeden komórki na 2 oddzielne części. W częściach tych zostały umieszczone przyciski Zapisz i Anuluj.

Pola:

- JLabel, JTextField, JButton – tak jak w PanelPacjenta
- JDateChooser – obiekt bedący rozwijanym kalendarzem. Po naciśnięciu na ikonkę w oknie, pojawia się aktualna data z kalendarzem na miesiąc. Obiekt pozwala na wybranie dowolnej daty i zapisuje ją w polu tekstowym.

Metody:

- SetKontroler, gettery i settery – tak jak w PanelPacjenta.
- UstawBadanie(Badanie) – metoda pobiera obiekt klasy Badanie i uzupełnia pola klasy PanelBadania wartościami obiektu Badanie.
- WyczyscBadanie – metoda czyści wszystkie pola klasy PanelBadania i ustawia je na wartość domyślną.

e) Panel Lista

Klasa będąca fragmentem interfejsu użytkownika, odpowiadająca za widok listy pacjentów (ich wyświetlanie), oparta na klasie JPanel. Poprzez pomocniczy model Listy, kompatybilny z tabelą JTable komunikuje się z modelem listy przechowującej pacjentów. Wywoływany za pomocą konstruktora, zawierającego odwołanie do prywatnej funkcji toGui.

Layout:

Do utworzenia tego panelu, nazwanego jako panelGlowny, zastosowany został GroupLayout. Składa się on z tabeli (typu JTable) oraz dwóch przycisków (rodzaju JButton) „Dodaj” oraz „Usuń”. JTable zostało umieszczone w panelu JScrollPane, żeby w razie potrzeby móc przewijać listę.

Pola:

- mTable – tabela prywatna typu JTable, potrzebna do GUI.
- mAddButton – przycisk prywatny typu JButton, odpowiadający za dodawanie nowego pacjenta, czyli uruchomienie panelu z danymi pacjenta.
- mRemoveButton – przycisk prywatny typu JButton, odpowiadający za usuwanie wybranego poprzez kliknięcie pacjenta.
- mKontroler – prywatna instancja Kontrolera.

Metody:

- toGui – metoda główna, uruchamiana poprzez konstruktor, tworzy i porządkuje cały Layout, zawiera panelGlowny – pole odpowiadające za wygląd całego panelu Listy oraz scroll – typu JScrollPane, do którego wrzucone jest mTable.
- selectRow - wybranie rzędu w mTable. Parametry wejściowe – numer rzędu.
- selectedRow - pobranie numeru wybranego rzędu z mTable.

- setKontroler - metoda rejestrująca kontroler, używana w „main” programu. Dodane są w niej listenery do mTable oraz do dwóch dostępnych w tej klasie przycisków mRemoveButton oraz mAddButton. Dostępna jest w niej również metoda valueChanged, umożliwiająca ciągłe nasłuchiwanie, czy któraś z linijek w liście nie została wybrana.
- odznacz - odznacza zaznaczenie z listy.
- setModelLista - podpina model listy do pomocniczego modelu z klasy PanelLista, umożliwiając współpracę modelu (listy, przechowującej pacjentów) z interfejsem JTable (mTable)
- odswiezJTable – odświeża widok (odświeża panel JTable).
- gettery i setery - automatycznie wygenerowane metody, pozwalające na ustawienie wartości prywatnych pól dla pacjenta oraz ich pobieranie.

Klasa pomocnicza MyTableModel:

Jest to klasa, umożliwiająca współpracę widoku z modelem. Ustawia ona liczbę i nazwy kolumn, wyświetla dane osobowe pacjentów, pobiera liczbę pacjentów w liście i umożliwia ich dodawanie oraz usuwanie. Dodatkowo, w jej wnętrzu tworzona jest lista, przechowująca dane pacjentów, pobrane od modelu – dzięki temu ich dane mogą być wyświetlanie na liście. Model ten jest modelem dla mTable.

f) Model Lista

Jest to klasa – model, przechowująca instancje pacjentów za pomocą listy rodzaju ArrayList. Ma ona za zadanie umożliwiać porządkowanie przechowywanych danych: usuwanie i dodawanie pacjentów z listy, dopinanie do pacjentów ich badań oraz pobranie danych wybranego pacjenta z listy.

Pola:

- lista – lista przechowująca pacjentów, rodzaju ArrayList

Metody:

- getValues - pobieranie prywatnej listy wartości (pacjentów)
- getPacjent - pobranie pojedynczego pacjenta z listy. Parametrem wejściowym jest numer indeksu z listy, numer zaznaczonego przez użytkownika pacjenta
- addPacjent - dodanie wartości do listy w modelu. Metoda działa w dwóch przypadkach – gdy żaden pacjent nie został wybrany z listy oraz gdy jeden pacjent na liście został zaznaczony. Gdy żaden pacjent nie był wybrany – sprawdzane jest, czy istnieje już osoba o danym numerze pesel, następnie sprawdzane jest, czy dana lista zawiera już identycznego pacjenta. W przypadku wybrania jednego z pacjentów, nowy pacjent powstaje na bazie starego – do nowego przypisywane jest badanie starego, który następnie jest usuwany, na jego miejsce wstawiany jest nowy ze zmienionymi przez użytkownika danymi. Dane wejściowe: pacjent i numer rzędu.
- removeValue - usuwanie wartości (pacjenta z rzędu r) z listy w modelu. Parametr wejściowy: numer rzędu wybranego pacjenta.
- przypiszBadanie - przypisanie badania do już istniejącego pacjenta. Dane wejściowe: badanie oraz numer rzędu wybranego pacjenta.

g) Kontroler

Kontroler to klasa umożliwiająca komunikację między widokiem a modelem. Jest ona implementowana poprzez ActionListener oraz FocusListener. Odpowiada ona za obsługę przycisków i wydarzeń.
Pola:
okno – pole prywatne klasy Okno
model – pole prywatne klasy ModelLista

Metody:

- addPacjent – dodaje pacjenta do modelu. Jako parametr wejściowy pobiera pacjenta. Połączone z metodą addPacjent z klasy ModelLista. Zwraca true albo false w zależności od tego, czy się wykona.
- removePacjent - usuwa pacjenta z modelu.
- addBadanie – dodawanie badania pacjenta do pacjenta obecnego w modelu. Jako parametr wejściowy pobiera pacjenta. Połączone z metodą przypiszBadanie z klasy ModelLista.
- removePacjent – usuwa pacjenta z modelu
- addBadanie – dodawanie badania pacjenta do modelu.
- wyswietlDane – wyświetlanie danych pacjenta oraz jego badania z wybranego rzędu z PanelLista w widoku. Parametr wejściowy – numer rzędu. Ustawia widoczność panelu Pacjenta na włączoną.
- zmienWidocznoscPacjenta – zmiana aktywności okna Pacjenta. Wartość wejściowa – true albo false
- zmienWidocznoscBadania – zmiana aktywności okna Badania. Wartość wejściowa – true albo false
- czySlowo - metoda pobiera łańcuch znaków i sprawdza czy każdy znak jest literą. Zwraca wartość logiczną true lub false.
- czyNumer - metoda pobiera łańcuch znaków i sprawdza czy każdy znak jest cyfrą. Zwraca wartość logiczną true lub false.
- panelPacjentZapisz - główna funkcja, zbierająca dane z pól tekstowych, sprawdzająca poprawność wpisanych danych i dodająca pacjenta. Najpierw odbywa się sprawdzenie poprawności wprowadzonych danych – sprawdzane jest, czy imię i nazwisko to słowa, nie zawierające innych znaków niż litery (dzięki metodzie czySlowo). Gdy wystąpi jeden z przypadków metoda ta sprawdza również inne możliwości błędów i w zależności od tego, które dane są źle podane wyświetla odpowiedni komunikat. Jeśli któreś z danych są niepoprawne, funkcja zwraca false. Jeśli dane są poprawne, funkcja ustawia ubezpieczenie oraz plec. Nastepnie pacjent jest dodawany (przy jednoczesnym sprawdzeniu, dzięki funkcji addPacjent z ModelLista, czy numer PESEL lub cały pacjent się nie powtórzył). Jeśli pacjenta nie udało się dodać, usuwane jest zaznaczenie z listy i zwracana jest wartość false. Jeśli pacjenta udało się dodać, zwracana jest wartość true.
- panelBadanieZapisz – dodawanie badania.
  Metody:
  - ggt, a1at, aspat- mówią o stężeniach pobranych od użytkownika; d – mówi o dacie pobranej od użytkownika
  - kol1 – odpowiada za kolor JTextFieldu GGT, domyślnie czarny, kol1 – odpowiada za kolor JTextFieldu A1AT, domyślnie czarny, kol1 – odpowiada za kolor JTextFieldu AspAT, domyślnie czarny
  - ret – zmienna logiczna, zmieniająca się zależności od tego, czy wystąpiły jakieś błędy podczas wykonywania poleceń.

  Najpierw dane są zbierane, przy jednoczesnym sprawdzeniu ich poprawności i złapaniu ich i obsłużeniu dzięki try-catch. Odpowiednie kolory są ustawiane na czerwony, w zależności od rodzaju występującego błędu, po czym każde z pól ma ustawiany kolor zmieniony na czerwony lub domyślnie czarny. Funkcja jest również wrażliwa na źle wprowadzoną datę oraz ma ograniczone wartości przyjmowane stężeń substancji (każda ze zmiennych może przymować wartości większe od zera i mniejsze od 100). Jeśli wystąpił któryś z wyjątków, funkcja nie jest dalej wykonywana – zwracana jest wartość false, jeśli natomiast dane są poprawne, dodawane jest badanie do pacjenta i zwracana wartość true.
- zmienKolor – funkcja zmieniająca kolor 3 JTextFieldów z panelu Badania. Parametry wejściowe to pożądane 3 kolory.

- actionPerformed – funkcja obsługująca przyciski nasłuchiwane w widokach. W zależności od przycisku wybranego przez użytkownika wykonują się odpowiednie polecenia.
    - zamknij – okno jest zamykane
    - zapisz (panel Pacjent) – zapisuje pacjenta do listy, gdy to się uda, wyłączany jest panel Pacjenta, a włączany panel Badania. Jeśli pacjenta nie udało się zapisać – panel Pacjenta jest dalej włączony, a panel Badania wyłączony.
    - anuluj (panel Pacjenta) – czyści pola tekstowe i pokazuje okno dialogowe „anuluj”
    - zapisz (panel Badania) – zapisuje badanie pacjenta do wybranego na liście pacjenta, jeśli to się uda, to panel Pacjenta dalej jest wyłączony, wyłącza się również panel Badania
    - anuluj (panel Badania) – ustawia na czarny kolor wszystkie pola tekstowe, wyłącza panel Badania i pokazuje okno dialogowe „anuluj”
    - getmAddButton (panel Lista) – pozwala dodać nowego pacjenta do listy, czyści dane badania i pacjenta, wyłącza panel Badania, a włącza panel Pacjenta. Odznacza zaznaczoną pozycję na liście.
    - getmRemoveButton (panel Lista) – pozwala usunąć pacjenta do listy, wyłącza panel Badania i panel Pacjenta. Odświeża widok mTable (poprzez funkcję odswiezJTable). Odznacza zaznaczoną pozycję na liście.
- focusGained – obsługa FocusEvent, dla każdego pola, które będzie aktywne pole tekstowe jest czyszczone.

h) Okno

Klasa Okno jest napisana na bazie klasy JFrame. Po wywołaniu obiektu klasy Okno na ekranie pojawia się ramka z odpowiednią zawartością. Ramka może być powiększana i w odpowiednich granicach zmniejszana bez straty wyświetlanej zawartości. Pasek na górze okna ma jedną opcją „Aplikacja”, która zamyka cały. Można skorzystać z skrótu klawiszowego alt+a i alt+z aby uzykać taki sam efekt.

Layout:

Layout ramki jest ustawiony na BorderLayout. Na pozycji „Center” ustawiony jest JPanel tlo, który ma kolor biały. JPanel tlo ma ustawiony layout na GridLayout. Dzięki temu, możemy podzielić go na 2 części – lewą i prawą. Lewa część jest dodatkowo podzielona na 2 mniejsze fragmenty, również za pomocą GridLayout. Do prawej części dodany jest obiekt PanelLista. Widoczność lewej części jest wyłączona, włącza się tylko po naciśnięciu przycisku dodaj w PanelLista.

Pola:

- PanelPacjenta – obiekt, który wypełnia lewą górną część okna. Służy do wpisywania danych pacjenta.
- PanelBadania – obiekt, który wypełnia lewą dolną część okna. Służy do wpisywania wyników badania.
- PanelLista – obiekt, który wypełnia prawą część okna. Służy podglądu dodanych pacjentów, dodawania nowych oraz usuwania.
- JPanel – obiekt, który jest elementem obiektu JFrame. Posiada swoje własne obramowanie z tytułem. Pozwala na dodanie do niego mniejszych elementów w odpowiednim układzie. Układ ustawiamy metodą setLayout().
- JMenuBar – obiekt, który znajduje się w górnej części okna. Do niego dołączone jest rozwijane menu.
- JMenu – obiekt, który jest dołączony do JMenuBar. Jest rozwijaną listą wyboru.
- JMenuItem – obiekt, który jest dodany do rozwijanej listy z paska menu.

Metody:

- set kontroler – tak jak w DanePacjenta.
- Gettery i settery – zwracają lub ustalają wartości obiektu danej klasy
- Widoczność Pacjenta, Badania – ustawiają panel na widoczny lub niewidoczny

i) Pokaż Okno

Plik tworzący obiekt klasy Okno, PanelPacjenta, PanelBadania, ModelLista. Tworzony jest także kontroler i każdy inny obiekt ustawia go, jako słuchacza zdarzeń.