### **1. Tytuł projektu**

System Zarządzania Kosztami Leczenia Pacjenta w Szpitalu

### **2. Opis projektu**

Program umożliwia obliczanie całkowitych kosztów leczenia pacjenta w szpitalu, w tym koszty procedur chirurgicznych, leków oraz pobytu w szpitalu. Zapewnia przyjazny interfejs użytkownika do zarządzania wyborem usług medycznych oraz generuje podsumowanie kosztów leczenia.

### **3. Skład zespołu**

Imię i nazwisko (Rola): Patryk Ruzicki (Developer)   
Imię i nazwisko (Rola): Krzysztof Łągiewczyk (Developer)   
Imię i nazwisko (Rola): Krystian Stróżczyk (Tester)

### **4. Cele projektu**

* Obliczanie kosztów leczenia pacjenta, w tym kosztów pobytu, zabiegów i leków.
* Zapewnienie przejrzystego interfejsu użytkownika.
* Umożliwienie dynamicznego wyboru zabiegów i leków.
* Generowanie podsumowania kosztów w łatwy do zrozumienia sposób.

### **5. Opis klas i metod**

#### **Klasa PatientAccount**

**Pola:**

* totalCost: całkowity koszt leczenia pacjenta.
* daysInHospital: liczba dni pobytu w szpitalu.
* dailyRate: stała określająca koszt jednego dnia pobytu (500 PLN).

**Metody:**

* PatientAccount(): konstruktor inicjalizujący pola.
* addCost(double amount): dodaje koszt do konta pacjenta.
* setDaysInHospital(int days): ustawia liczbę dni pobytu.
* calculateHospitalStayCost() const: oblicza koszt pobytu.
* getTotalCost() const: zwraca całkowity koszt leczenia.

#### **Klasa Surgery**

**Pola:**

* Operacje: lista dostępnych procedur chirurgicznych.

**Metody:**

* Surgery(): konstruktor inicjalizujący listę procedur.
* performSurgery(PatientAccount& patient, int choice): dodaje koszt wybranego zabiegu do konta pacjenta.
* displaySurgeries(): wyświetla listę dostępnych zabiegów.

#### **Klasa Pharmacy**

**Pola:**

* medicines: lista dostępnych leków.

**Metody:**

* Pharmacy(): konstruktor inicjalizujący listę leków.
* prescribeMedicine(PatientAccount& patient, int choice): dodaje koszt wybranego leku do konta pacjenta.
* showMedicines(): wyświetla listę dostępnych leków.

### **6. Struktura kodu**

* main.cpp: Główny plik programu zawierający logikę menu i interakcji z użytkownikiem.
* patientAccount.h / patientAccount.cpp: Definicja i implementacja klasy PatientAccount.
* Surgery.h / Surgery.cpp: Definicja i implementacja klasy Surgery.
* Pharmacy.h / Pharmacy.cpp: Definicja i implementacja klasy Pharmacy.

### **7. Opis działania programu**

Użytkownik:

1. Podaje liczbę dni pobytu w szpitalu.
2. Może wybrać procedury chirurgiczne z listy i dodać je do konta pacjenta.
3. Może wybrać leki z listy i dodać ich koszt do konta pacjenta.
4. Po zakończeniu wybierania użytkownik otrzymuje podsumowanie kosztów leczenia, które uwzględnia pobyt, zabiegi i leki.

### **8. Problemy i rozwiązania**

* **Problem 1:** Obsługa błędnych danych wejściowych (np. wybór opcji spoza zakresu).
  + **Rozwiązanie:** Dodano walidację wejścia z komunikatami o błędach.
* **Problem 2:** Brak możliwości dynamicznej edycji listy zabiegów i leków.
  + **Rozwiązanie:** Użycie wektorów do przechowywania danych, co umożliwia łatwe dodawanie i modyfikację.

### **9. Postęp projektu**

Projekt przebiegał zgodnie z harmonogramem. Udało się zaimplementować wszystkie główne funkcjonalności. Najwięcej czasu pochłonęło dopracowanie interfejsu użytkownika oraz testy poprawności obliczeń.

### **10. Wnioski i przyszłe usprawnienia**

* Możliwość zapisania podsumowania kosztów do pliku.
* Dodanie opcji usuwania wybranych zabiegów lub leków z konta pacjenta.
* Wprowadzenie graficznego interfejsu użytkownika.
* Integracja z bazą danych do przechowywania historii pacjentów.

Podpisy zespołu: Patryk Ruzicki, Krzysztof Łągiewczyk, Krystian Stróżczyk