

*Postęp nie był dokumentowany w platformie GitHub, jednak kolejne elementy projektu powstawały zgodnie z założonym harmonogramem zajęć laboratoryjnych, aktualizowane z tygodnia na tydzień.*

### **Tydzień 1 (21.10 - 27.10):**

- utworzenie pierwszych dwóch plików: "pilka.h" oraz "paletka.h"
- początkowe utworzenie klasy "pilka" wraz z jej polami prywatnymi "x", "y", "vx", "vy", "r" oraz metodami publicznymi: "przesun", "odbijx", "odbijy", "kolizjaSciana" oraz pobierającymi dane: "getX", "getY", "getVX", "getVY"
- utworzenie zarysu klasy paletka z polami prywatnymi "x", "y", "vy", "szerokosc", "dlugosc"

### **Tydzień 2 (28.10 - 03.11):**

- przygotowanie środowiska do pracy z SFML

### **Tydzień 3 (04.11 - 10.11):**

- dopracowanie oraz edycja pól klasy "pilka":
  - pola prywatne: "r" zamieniono na "radius", dodano pole "sf::CircleShape ball"
  - konstruktor: ustawia pozycję startową, prędkość w obu kierunkach i promień, ustawia: promień okręgu, punkt odniesienia w centrum, pozycję oraz kolor wypełnienia
  - pola publiczne: zamieniono ze sobą (i edytowane) następujące funkcje: "przesun" -> "move", "odbijx" -> "bounceX", "odbijy" -> "bounceY" oraz "kolizjaSciana" -> "collideWalls"; dodano funkcje: "collidePaddle", "draw" oraz funkcje pobierające dane: "getX", "getY", "getVX", "getVY" oraz "getRadius"
- dopracowanie oraz edycja pól klasy "paletka":
  - pola prywatne: "vy" zmieniono na "predkosc", dodano pole "sf::RectangleShape pallet"
  - konstruktor: ustawia pozycję startową, szerokość, wysokość i prędkość, ustawia: wielkość, punkt odniesienia w centrum, pozycję oraz kolor wypełnienia
  - pola publiczne: dodano funkcje "moveLeft", "moveRight", "clampToBounds" oraz "draw"; dodano funkcje pobierające dane: "getX", "getY", "getSzerokosc", "getWysokosc"
- stworzenie pliku "main.cpp" zawierającego pętlę główną programu; tworzone jest okno gry i jej obiekty, a główna pętla: obsługuje zdarzenia i sterowanie klawiatury, zgodnie z zasadami wersji 2.6 SFML, aktualizuje pozycję piłki, kontroluje przegraną, rysuje obiekty a dodatkowo: co kilka klatek wypisuje w konsoli parametry piłki, w momencie uderzenia piłki o paletkę w konsoli wyświetlany jest komunikat "HIT PADDLE", w momencie przegranej konsola wyświetla "MISS! KONIEC GRY"

### **Tydzień 4 (12.11 - 17.11):**

- dodanie pliku “kamien.h” oraz “kamien.cpp”:
  - utworzenie klasy kamień dziedziczącej po sf::RectangleShape w pliku “kamien.h”
  - dodanie pól prywatnych klasy: “m\_punktyZycia”, “m\_jestZniszczony” oraz “static const std::array<sf::Color, 4> m\_colorLUT”
  - dodanie konstruktora, który: inicjalizuje początkowe HP bloków, wywołuje metody określające pozycję oraz rozmiar, ustawia obrys oraz aktualizuje kolory poszczególnych bloków
  - dodanie metod publicznych: “trafienie”, “aktualizujKolor”, “isDestroyed” oraz “draw”
  - w pliku “kamien.cpp” utworzono statyczną tablicę kolorów dla bloków z poszczególną ilością punktów życia
- zmiany w pliku “main.cpp”:
  - utworzenie pustego wektora na bloki
  - zdefiniowanie stałych pozwalających na stworzenie stałych: “ILOSC\_KOLUMN”, “ILOSC\_WIERSZY”, “ROZMIAR\_BLOKU\_Y”, “ROZMIAR\_BLOKU\_X”
  - zdefiniowanie pętli generujących poziom

### **Tydzień 5 (18.11 - 24.11):**

- utworzenie plików “gra.h” oraz “gra.cpp”:
  - stworzenie klasy “gra” z polami prywatnymi: “paletka m\_paletka”, “paletka m\_pilka”, “std::vector<kamien> m\_bloki” oraz metodami kontrolującymi grę (aktualizacja elementów, generowanie poziomu)
- ogólne uporządkowanie kodu i segregacja na pliki nagłówkowe oraz źródłowe
- w pliku main.cpp: zdefiniowanie stanów, generowanie okna, obsługa poszczególnych stanów gry
- utworzenie plików “main.h” oraz “main.cpp” do rysowania menu wraz z metodami pozwalającymi na jego obsługę

### **Tydzień 6 (25.11 - 01.12):**

- utworzenie plików “aktualnyStanGry.h” oraz “aktualnyStanGry.cpp” zawierające aktualne pozycje elementów, wraz z metodami pozwalającymi na odczyt, zapis a także załadowanie gry z zapisu
- dodanie obsługi pliku tekstowego do używania zapisu gry oraz opcji wczytania z zapisu z menu

### **Tydzień 7 (02.12 - 10.12):**

- dopracowanie pojedynczych elementów gry
- dodanie plików “kryształ.h” oraz “kryształ.cpp” z obsługą losowo pojawiających się, nieregularnych kształtów do zbierania
- dodanie tekstowego elementu zliczającego ilość zdobytych punktów

**STEROWANIE:**

- sterowanie paletką strzałkami prawo/lewo
- sterowanie opcjami menu strzałki góra/dół, wybieranie opcji przyciskiem enter
- zapis gry przyciskiem P z sekundową przerwą