Zadanie 2: Gra w odcinki 2D
(część do oddania na pracowniach 16-21.X, wersja ostateczną 23-28.X) **20 punktów**:

Napisać zgodnie z poniższymi wymaganiami implementacyjnymi prostą 2D grę w odcinki. Celem gry jest zmieszczenie jak największej ilości nieprzecinających się odcinków lub trójkątów na planszy (w oknie). Gracz przy każdym kliknięciu myszą dodaje odcinek którego jednym końcem jest położenie myszy, a drugi jest wybierany losowo tak aby długość odcinka była nie większa niż wybrana stała np. 1/10 długości boku okna. Ostatni odcinek warto wyróżnić chwilowo rysując go inaczej np. dorysowując dodatkowo końce, lub kilka równoległych linii. Kolejne odcinki rysujemy różnymi kolorami np. naprzemiennie 3 z ustalonej tablicy (np. można to zrobić w vertex shaderze, wybierając kolor na podstawie indeksu wierzchołka gl_VertexID).

Gra kończy się przy pierwszej lub k-tej kolizji (wartość domyślną dobrać dla grywalności, można ją zmienić klawiszami 1,2,3,4,5). Po skończeniu gry można zamknąć okno naciśnięciem klawisza Esc lub klawiszem 'n' zacząć nową grę (oba klawisze powinny też działać w trakcie grania).

Miedzy wariantami gry wybieramy klawiszami 'o' i 't'. W drugim wariancie gry zamiast odcinków dodajemy trójkąty prostokątne równoramienne, tak aby wierzchołek z kątem prostym był w kliknięty punkcie, a kierunek/wzór trójkąta możemy wylosować spośród 4 lub więcej wzorów trójkątów.

Program powinien pokazywać aktualną ilość odcinków/trójkątów czyli punkty, także końcowe. Wyniki te można wypisywać na terminalu oraz wizualizować w postaci dwóch słupków/odcinków w różnych kolorach na prawym skraju okna (wynik maksymalny to połowa wysokości okna i obok słupek z wynikiem bieżącym).

Wymagania implementacyjne:

Można bazować np. na 2 tutorialu link. Rysujemy plansze 2D więc nigdzie nie ma trzeciego wymiaru, tak więc nie można używać macierzy przekształceń 3D. Wierzchołki maja tylko 2 współrzędne 2D! Wierzchołki pamiętamy w układzie współrzędnych NDC (Normalized Device Coordinates) czyli [-1,1]x[-1,1]. Przy pomocy funkcji glfwSetWindowSizeCallback() zadbać aby okno właściwie reagowało na zmianę rozmiaru (handler powinien zaktualizować przekształcenie z NDC do współrzędnych w pikselach).

Ocenianie i dodatkowe funkcjonalności:

- 3pkt - 16-21.X oddanie prostej wersji na zajęciach dla odcinków, kolizje mogą nie działać poprawnie
- 11pkt - 23-28.X wersja finalna bez poniższych dodatków
- 3pkt - dodanie ładnego tła renderowanego przy pomocy jednego prostokąta i dodatkowego fragment shadera, generującego zmienne kolory typu fale, szachownice,... (dobrane tak aby nie popsuć widoczności obiektów)
- 3pkt - efekt końcowy w postaci animacji typu np. cała plansza kurczy się do punktu na środku okna (lub przewraca się 2D!) i jednocześnie zmienia kolory, które stopniowo stają się bardziej wypłowiałe/szare lub bardziej czerwone. Efekt należy uzyskać używając uniformów typu float (wielu lub jednego), które przekazujemy do shadera (np. dla skalowania, dla mieszania dwóch kolorów).