Zadania do wykonania (1)

Na ocenę **3.0** należy nacieszyć oko przykładowym program i uruchomić go po wprowadzeniu drobnych zmian w kodach shaderów.

Wskazówki:

- dla potwierdzenia, proszę zmienić kolor wyświetlanego sześcianu,
- utworzyć nową zmienną **wyjściową** w shaderze wierzchołków,
  - `out vec4 vertex_color;`
- w funkcji `main()` shadera wierzchołków nadać jej wartość, np.
  - `vertex_color = vec4(0.2, 0.9, 0.1, 1.0);`
- utworzyć nową zmienną **wejściową** w shaderze fragmentów,
  - `in vec4 vertex_color;`
  - nazwa musi się pokrywać z wyjściem shadera wierzchołków (!),
- w funkcji `main()` shadera fragmentów przypisać przekazaną wartość,
  - `color = vertex_color;`

Politechnika Wrocławska

Zadania do wykonania (2)

Na ocenę **3.5** należy zmodyfikować kolory bryły w przykładowym programie.

Wskazówki:

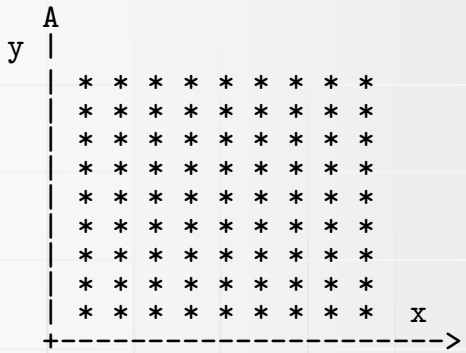
- celem jest aby każdy bok przykładowego sześcianu miał inny kolor,
- konieczne będzie zadeklarowanie dodatkowej zmiennej wejściowej (**in**) w shaderze wierzchołków oraz przekazanie jej do shadera fragmentów,
- zadanie można rozwiązać na dwa sposoby; konieczne będzie:
  - rozszerzenie tablicy **vertex_positions** o wartości kolorów, lub
  - zdefiniowanie nowej tablicy z kolorami oraz bufora danych;
- jeśli wybrano wariant z roszerzeniem tablicy **vertex_positions**:
  - zmodyfikować pierwsze wywołanie **glVertexAttribPointer()**, aby uwzględnić przesunięcie kolejnych informacji w tablicy z danymi;
- przekazać drugą tablicę na wejściowe shadera wierzchołków, dodając nowe wywołania **glVertexAttribPointer()** oraz **glEnableVertexAttribArray()**.

Zadania do wykonania (3)

Na ocenę **4.0** należy stworzyć wiele kopii obiektu (klasycznie, na CPU).

Wskazówki:

- Należy utworzyć planszę, złożoną z wielu instancji, np. 10 x 10.
- Obiekty można rozmieścić, stosując translację wzdłuż osi X i Y.
- Cała zmiana powinna ograniczyć się wyłącznie do funkcji `render()`,
  - slajd 18 zawiera prawie wszystkie niezbędne elementy.
- Konieczne może być oddalenie kamery (**V_matrix**), aby zobaczyć efekt.
- Poglądowy widok na tworzoną scenę:



Zadania do wykonania (4)

Na ocenę **4.5** należy wykorzystać mechanizm renderowania instancyjnego.

Wskazówki:

- Celem jest uzyskanie takiego samego efektu, jak w poprzednim zadaniu,
  - tym razem implementacja będzie znacznie wydajniejsza,
  - większość zmian obejmie kod shadera wierzchołków.
- Aby przetransformować cały obiekt (wszystkie jego wierzchołki tak samo), należy uzależnić transformacje wierzchołków od zmiennej **gl_InstanceID**.
  - Przydatna będzie funkcja moduł (%).
- Można zaimplementować funkcję, realizującą odpowiednią transformację.

Zadania do wykonania (5)

Na ocenę **5.0** należy wprowadzić dodatkowe deformacje każdego obiektu.

Wskazówki:

- Deformacje zrealizować na poziomie shadera wierzchołków.
- Warto wykorzystać wykorzystać funkcje pseudolosowe do transformacji.
- W języku GLSL nie ma standardowo dostępnej funkcji **rand()** – należy ją zdefiniować jako dowolny ze znanych generatorów pseudolosowych.
- Przykładowe funkcje: https://en.wikipedia.org/wiki/List_of_random_number_generators.
- Uzależnić transformacje od zmiennej wbudowanej **gl_VertexID**.
- Uwzględnić także **gl_InstanceID**, aby każdy obiekt deformować inaczej.