Zbiór Zadań
Praktyczny Wstęp do Programowania
część 2.

Maciej Matyka (maciej.matyka@uwr.edu.pl)

Strona FB kursu:
https://www.facebook.com/PWDPWFA/

Zasady Zaliczeń:
http://www.ift.uni.wroc.pl/~maq/zajecia/wp2017z/zasady2018.pdf

Skala ocen:
130 bdb
118 db+
105 db
93 dst+
80 dst

- Jest około 30 zadań (mamy też zadania dodatkowe, ich liczba się zmienia).
- Zadania 1-20 należy wykonać je **w pierwszej połowie semestru**.
- Zadania 9, 19 oraz 24 **są obowiązkowe**
- Zadania **wykonujemy samodzielnie**
  (proszę ignorować wszelkie “pomoce” z poprzednich lat)

Zbiór zadań do kursu

21. Ciąg Fibonacciego

Sprawdź, czy poprawnie zapisałem 45. w kolejności liczbę Fibonacciego (1134903170). Użyj wzoru rekurencyjnego (internet) i zapisz odpowiednią funkcję.
Uwaga: zadanie jest „trywialne” w tym sensie, że można łatwo znaleźć rozwiązanie w sieci (Google). Proszę podejść ambitnie i spróbować je rozwiązać samodzielnie. Ważne jest to, aby napisać funkcję F(n) w sposób rekurencyjny.
(5 pkt za samodzielne rozwiązanie, 1pkt za znalezione w sieci)

22. SFML

Przepisz/skopiuj/ściągnij kod keysfml.cpp z wykładu

```
#include <SFML/Window.hpp>
#include <SFML/Graphics.hpp>

int main()
{
        int keyonoff=1;

        sf::RenderWindow window(sf::VideoMode(800, 600), "Nasze okno");
        sf::CircleShape shape(400.f);
        shape.setFillColor(sf::Color::Red);

        while (window.isOpen())         // dopoki okno jest otwarte...
        {

         // w tym obiekcie klasy sf::Event
         // bedziemy mieli informacje co zrobil uzytkownik
        sf::Event event;

        while (window.pollEvent(event))
        {
        // uzytkownik kliknal zamkniecie  okna
        if (event.type == sf::Event::Closed)
                        window.close();

                // uzytkownik nacisnal klawisz
                if(event.type== sf::Event::KeyPressed)
                        keyonoff=1-keyonoff;
   }

        // czyszczenie (na czarno)
   window.clear(sf::Color::Black);


   // rysuj kolo w zaleznosci od stanu zmiennej keyonoff
   if(keyonoff)
        window.draw(shape);

        window.display();
        }
        return 0;
}
```

Program skompiluj i uruchom, wykorzystaj wiadomości ze strony:
http://www.sfml-dev.org/tutorials/2.3/ (Getting Started)
(3 pkt)

23. Zabawa SFML

Dokonaj modyfikacji w poprzednim zadaniu:

1) Kółko jest za duże, zmień jego rozmiar tak by zajmowało ¼ ekranu. (1 pkt)
2) Szerokość i wysokość okna oraz promień koła umieść w stałych. Następnie z tych wielkości wylicz, gdzie powinien się znajdować początek aby koło było wycentrowane. (1 pkt)
3) Wprowadź zmienne „x" i „y" typu float oznaczające pozycję koła i wykorzystaj funkcję setPosition aby je odpowiednio ustawić: http://www.sfml-dev.org/tutorials/2.3/graphics-transform.php (1 pkt)
4) Rozszerz program tak, aby zamykał okno po nacśnięciu „Esc". Wskazówka: kody klawiszy znajdują się w dokumentacji api biblioteki SFML (1 pkt)
5) Dodaj do programu zmienne „vx" i „vy" typu float oznaczające prędkość koła. Początkowe wartości mogą być np. 0.05. Dopisz przesuwanie koła: x=x+vx*dt, y=y+vy*dt. (1 pkt)
6) Uwzględnij, że koło może wyjść poza ekran i w takich przypadkach odbij prędkość zgodnie z zasadą, że ruch w kierunku „x" jest niezależny od ruchu w kierunku „y". (1 pkt)
7) Wprowadź stałą „N" oznaczającą ilość elementów. Wykorzystaj tablice jednowymiarowe (wprost zamień zmienne x,y,vx oraz vy na tablice) oraz dodaj tablicę kształtów „shapes" tak, aby obsłużyć N poruszających się kółek, zamiast jednego.
   (2 pkt)
8) Dodaj grawitację, która zmieni w każdym kroku prędkość poruszających się kół (może być najprostsza metoda Eulera) tak, aby punkty odbijały się i poruszały po parabolach, zamiast po prostych.
   Możesz to zrobić np. wykonując:
   x=x+vx * dt
   y=y+vy * dt
   vy = vy + g*dt // grawitacja
   (2 pkt)
9) Dopisz oddziaływanie N^2, dopisz uzależnienie koloru od prędkości ciał, dopisz kolizje pomiędzy ciałami itp. etc. (2 pkt)

(max 12 pkt)

24. Projekt GRA (obowiązkowe)

Został tydzień do premiery systemu operacyjnego Windows. Zostałeś właśnie wezwany do Billa Gates'a, który wskazał, że w jego nowym produkcie brakuje elementu rozrywkowego – gry, która będzie prosta i pozwoli użytkownikom na chwilę relaksu w trakcie ciężkiej pracy.

Wymyśl swoją grę (lub zaimplementuj klasycznego sapera), która posiada następujące cechy:

a) Gra jest grą - zdobywamy punkty, mamy cel, możemy zginąć w grze, gra wciąga (10 pkt)
b) Stan gry/mapa/układy muszą być przechowywane w tablicach 1D/2D - (3 pkt)
c) Gra zawiera element losowy (przy uruchomieniu różny układ przeszkód / korytarzy etc) - (3 pkt)
d) Gra powinna posiadać ładną wizualizację stanu gry (np. SFML lub tryb ANSI) (5 pkt)
e) Gra umożliwia zapisanie i odczytanie stanu do pliku tekstowego (2 pkt)
f) Kod gry jest strukturalny - używa funkcji, przekazuje parametry, nie ma zmiennych globalnych (4 pkt)
g) Gra będzie oryginalna (niespotykana w sieci, etc.) (3 pkt)
(max 30pkt)

25. Rysunek

Napisz program, który generuje dowolny, losowy rysunek w jeden ze sposobów:
- losowo rozrzucone koła, kwadraty i linie
- grafika wygenerowana innym (np. własnym) algorytmem
Uwaga: program powinien być wykonany z użyciem funkcji (oczywiście wg uznania) , np. koło z podaniem promienia, pozycji oraz adresu tablicy z rysunkiem gdzie ma być wrysowane.
Wskazówka: Program powinien trzymać w pamięci rysunek w postaci tablicy dwuwymiarowej struktur RGB:
struct RGB rysunek[H][W];
gdzie H to wysokość rysunku, W jego szerokość, a struktura RGB zdefiniowana jest jako:

```
struct RGB
{
        int R;
        int G;
        int B;
};
```

(7pkt)

26. Zapis .ppm

Dopisz do programu z zad. 1 funkcję saveppm, która pobierze nazwę pliku jako char * lub string oraz tablicę rysunek i zapisze te dane rysunku na dysku w formacie .ppm.
(3pkt)

27. Animacja

Wykorzystaj zadanie 26 i wygeneruj kilka klatek animacji, tzn. plików w formacie .ppm, nazwanych np. frame001.ppm, frame002.ppm itd. które przedstawiają ruch jakiegoś obiektu (np. jednego z kół). Tu można popuścić wodze fantazji i pobawić się trochę ruchem obiektów, zachęcam do eksperymentów! (4 pkt)
Klatki przekonwertuj do postaci animacji .mpg lub .avi (jest na to wiele sposobów, proszę próbować samodzielnie lub skorzystać z programów: ppmtojpg i avconv pod linux lub IrfanView i VirtualDub pod Windows).

Animację (lub link do pliku animacji w youtube/vimeo/FB) wyślij emailem do prowadzącego kurs: maciej.matyka@uwr.edu.pl (4pkt)
Wskazówka: do generowania nazw plików można użyć np. stringstreams: https://cs50.harvard.edu/resources/cppreference.com/cppsstream/all.html
(max 8pkt)

28. Fraktal

1.1) Zaimplementuj algorytm z Wikipedii i wyrysuj fraktal ze zbioru Mandelbrota dla kilku ustawień parametrów:

```
Dla każdego piksla rysunku (Px, Py) wykonaj:
{
        x0 = przeskalowana współrzędna x (np. leżąca w przedziale (-2.5, 1))
        y0 = przeskalowana współrzędna y (np. leżąca w przedziale (-1, 1))
        x = 0.0
        y = 0.0
        iteracja = 0
        max_iteracji = 1000
        dopóki ( x*x + y*y < 2*2 ORAZ iteracja < max_iteracji )
        {
                xtemp = x*x - y*y + x0
                y = 2*x*y + y0
                x = xtemp
                iteracja = iteracja + 1
        }
        kolor = alfa * (iteracja / max_iteracji)
        narysuj pixel w pozycji (Px, Py) o kolorze kolor
        }
```

Uwaga: współczynnik alfa proszę dobrać eksperymentalnie.
(źródło: http://en.wikipedia.org/wiki/Mandelbrot_set#Escape_time_algorithm)

Obraz zapisz na dysku w formacie PPM.
(7pkt)

29. Animowany fraktal

Dopisz do swojego programu możliwość wygenerowania klatek (dla zmieniającego się parametru zbioru Mandelbrot-a). Animację wyślij na YouTube, profil Facebook i poinformuj wykładowcę (maciej.matyka@uwr.edu.pl).
(5 pkt)

Zadania dodatkowe (2017)

Zadania, które nie zwiększają puli punktów obowiązkowych do zrobienia, ale pozwalają zdobyć ekstra punkty i nie wykonywać standardowych zadań.

30. Własny projekt

Wykorzystaj wiedzę z kursu do stworzenia programu, gry, rysunku lub animacji wg własnego oryginalnego pomysłu. Pomysł skonsultuj z wykładowcą.
(40pkt)

31. Konsola 24 bit

Wygeneruj fraktal, gradient kołowy lub inny ciekawy wykres w postaci mapy kolorów w konsoli w trybie 24-bitowym (liczy się efekt).
(3pkt)

32. Animacja w konsoli

Korzystając z wiedzy nt kodów ANSI oraz funkcji Sleep/usleep (w zależności Windows/Linux) napisz prostą animację w konsoli (podobnie do zadania 9, ale albo generując funkcyjnie kolejne klatki albo wyświetlając kilka klatek w pętli).
(3pkt)

Literatura do całego kursu (książka jest dostępna w czytelni WFiA):
Brian W. Kernighan, Dennis M. Ritchie, Język C Programowanie, Wydanie 2, Helion 2010
http://wiki.fedora.pl/wiki/Kod_ANSI