Projekt 1 – Grafika wektorowa

Celem projektu jest skonstruowanie szkieletu aplikacji pozwalającej na tworzenie grafiki wektorowej w sposób zbliżony do popularnych programów typu CorelDraw, Adobe Illustrator, Shockwave Flash, czy AutoCAD. Grafika wektorowa w przeciwieństwie do grafiki rastrowej, gdzie obraz zapamiętywany jest w postaci siatki pikseli zawierającej kolor każdego piksela w postaci składowych RGB, do opisu obrazu wykorzystuje bardziej złożone obiekty postaci linia, wielokąt, elipsa, itp. (patrz rys. 1).

Rys. 1. Wektorowy obraz twarzy wyrysowany przy pomocy pięciu elips

Takie rozwiązanie pozwala na znaczną redukcję długości plików przy jednoczesnych zachowaniu poziomu szczegółowości, wystarczającym w wielu przypadkach do przekazania niezbędnej informacji. Państwa zadanie zakłada oczywiście pewne uproszczenia i koncentrować się będzie jedynie na konstrukcji szkieletu umożliwiającego przechowywanie składowych obiektów obrazu oraz wykonywanie na nich prostych przekształceń. Do wspomnianych uproszczeń należą:

- Rysunki mają być wykonywane w trybie tekstowym*,
- Do grupy obiektów składowych obrazu zaliczamy: punkt, linię, kwadrat i elipsę,
- Należy zaimplementować jedynie przykładowe operacje typu skalowanie, odbicie w pionie czy w poziomie.

Aplikacja powinna zawierać dodatkowo obiekty:

- **Obraz,** służy do przechowywania wszystkich obiektów wchodzących w skład grafiki wektorowej.
- **Ekran,** zawiera wynik otrzymany po wyrysowaniu wszystkich obiektów. W realizowanym przypadku będzie to oczywiście tablica znaków.
- **Przekształcenie**, pozwala na wykonywanie wspomnianych operacji na elementach składowych obrazu.

Użytkownik programu powinien mieć możliwość dodawania do obrazu nowych obiektów oraz definiowania ich parametrów, a także możliwość przyłączania i odłączania od obiektów już istniejących, wybranych przez siebie przekształceń. Ponadto należy użytkownikowi umożliwić zapis obrazu w postaci tekstu opisującego obiekty składowe i przyporządkowane im przekształcenia oraz w postaci zrzutu ekranu tekstowego. Wraz z programem należy oddać sprawozdanie, którego szablon będzie wkrótce dostępny.

*Dla osób zainteresowanych udostępniam bibliotekę do wizualizacji obrazów w trybie graficznym (patrz druga strona).

Biblioteka graficzna.

Dla osób zainteresowanych wyświetlaniem obrazów w trybie graficznym proponuję specjalną bibliotekę, która udostępnia funkcje rysujące obiekty typu: punkt, linia, wielokąt oraz elipsa. Obiekty te wyrysowywane są w automatycznie otwierającym się oknie aplikacji Windows API. Poniżej znajduje się dokładny opis dostępnych funkcji oraz przykładowy program rysujący „twarz" (z rysunku 1).

Opis funkcji:

```
void UstawRozmiar(int szer, int wys);
```

Funkcja ustala szerokość (szer) i wysokość (wys) części okna przeznaczonej do rysowania.

```
void Czysc(KOLOR k);
```

Funkcja czyści okno zapisując jego zawartość jednolitym tłem w zadanym kolorze k, gdzie KOLOR to struktura definiowana jako

```
struct KOLOR
{
    int         red;
    int         green;
    int         blue;
};
```

zawierające składowe RGB koloru.

```
void ZmienNazwe(string nazwa);
```

Funkcja ta ustala nazwę okna podaną jako obiekt klasy string.

```
void UstawPunkt(int x, int y, KOLOR k);
```

Funkcja rysuje piksel w punkcie o współrzędnych x i y i w zadanym kolorze k.

```
void RysujElipse(int x0, int y0, int x1, int y1, KOLOR k_ramki, KOLOR k_tla);
```

Funkcja rysuje elipsę o kolorze wypełnienia k_tla oraz kolorze ramki k_ramki, wpisaną w prostokąt opisany za pomocą współrzędnych górnego lewego rogu (x0, y0) oraz prawego dolnego rogu (x1, y1).

```
void RysujWielokat(int liczba, PUNKT* punkty, KOLOR k_ramki, KOLOR k_tla);
```

Funkcja rysuje wielokąt w kolorze wypełnienia k_tla i ramce w kolorze k_ramki, opisany za pomocą punktów (wierzchołków), których współrzędne zapisane są w liczba- elementowej tablicy punkty obiektów typu PUNKT

```
struct PUNKT
{
    int         x;
    int         y;
};
```

```
void RysujLinie(int x0, int y0, int x1, int y1, KOLOR k);
```

Funkcja rysuje linię w kolorze k od punktu o współrzędnych (x0, y0) do punktu (x1, y1).

Biblioteka graficzna składa się z trzech plików, które muszą się znaleźć w katalogu tworzonego projektu.

Okno.h – plik nagłówkowy, który należy dołączyć do pliku aplikacji za pomocą dyrektywy `#include "Okno.h"`.

Okno.o – plik ten należy dołączyć do projektu poprzez zakładkę: Projekt -> Opcje projektu -> Parametry -> Dodaj plik.

wndcon.ovl – plik aplikacji okienkowej Windows API.

Przykładowy program:

Zamieszczony poniżej program jest dostępny na stronie w postaci projektu skompresowanego do pliku typu RAR.

```
#include <cstdlib>
#include <iostream>
#include <conio.h>
#include "Okno.h"

using namespace std;

int main(int argc, char *argv[])
{
    KOLOR    szary, zolty, niebieski;
    Okno     okno(256, 256, "Aplikacja przykladowa - Twarz");
    szary.red     = 220; szary.green     = 255; szary.blue     = 220;
    zolty.red     = 255; zolty.green     = 255; zolty.blue     = 128;
    niebieski.red = 128; niebieski.green = 128; niebieski.blue = 255;
    okno.Czysc(szary);
    okno.RysujElipse(32, 32, 224, 224, niebieski, niebieski);
    okno.RysujElipse(64, 160, 192, 196, zolty, zolty);
    okno.RysujElipse(64, 140, 192, 176, niebieski, niebieski);
    okno.RysujElipse(64, 64, 96, 128, zolty, zolty);
    okno.RysujElipse(160, 64, 192, 128, zolty, zolty);
    system("PAUSE");
    return EXIT_SUCCESS;
}
```

Listing 1. Kod programu Twarz.