Przetwarzanie obrazów

Na chwilę porzucimy elementy interfejsu użytkownika i zajmiemy się innymi, może nawet bardziej interesującymi zastosowaniami programowania. Jednym z nich jest przetwarzanie obrazów.
Kod przetwarzania obrazu napiszemy w języku C – będzie to jedna prosta funkcja wykonująca filtrację obrazu. Żeby nie komplikować niepotrzebnie sprawy, ograniczymy się (w tym zadaniu) do obrazów RGB: takich, w których jeden piksel jest reprezentowany przez trzy bajty odpowiadające składowym czerwonej, zielonej i niebieskiej koloru.
Prócz operowania na obrazach w realizacji tego zadania pojawi się więcej nowości:

- operowanie na plikach binarnych,
- wykorzystanie kodu napisanego w C,
- przekazywanie parametrów do programu wykonywalnego.

Parametry programu

W linii komend możemy podać parametry do uruchomienia programu:

```
C:\....\zad_7\bin\Debug>zad_7 tiger.bmp tiger_out.bmp
```

Żeby mieć dostęp do tych parametrów deklarujemy funkcję `main` w następujący sposób:

```
int main(int argc, char* argv[])
```

`argc` oznacza liczbę argumentów (zawsze `>= 1`, bo pierwszym argumentem jest nazwa uruchamianego programu).

Dostęp do kolejnych argumentów (w postaci ciągów znaków zakończonych zerem) dają kolejne elementy tablicy `argv`.

Problem w tym, że uruchamiając program w `Code::Blocks` nie widzę miejsca na argumenty (można je ustawić w `Project > Set programs' arguments`).

Operacje na obrazie

Szczęśliwie czytanie i zapis plików obrazów RGB w formacie BMP jest już zaimplementowane.

Do zaimplementowania są funkcje `get_pixel` i `put_pixel` i trzeba tutaj przeliczyć współrzędne `(x, y)` piksela na offset (indeks) względem początku obrazu:

```
offset = y * linebytes + x * 3
```

Drugą operacją, wykonywaną niezależnie dla każdego piksela, jest filtr sepii. Nowe wartości składowych wyznaczamy na postawie składowych aktualnych w następujący sposób:

```
new_red = red * 0.393 + green * 0.769 + blue * 0.189
new_green = red * 0.349 + green * 0.686 + blue * 0.168
new_blue = red * 0.272 + green * 0.534 + blue * 0.131
```

Jest tu jeszcze haczyk: wartość może wyjść większa niż 255 – trzeba ją wtedy ustawić na maksymalne 255.

Wynik filtrowania

Oddawanie

Rozwiązanie zadania składa się z 3 plików:

`image.h` – zawiera definicje struktur oraz deklaracje funkcji

`image.c` – implementacje funkcji

`zad_7.c` – funkcja `main` w postaci z ćwiczeń.

Rozwiązanie proszę załadować do `Moodle` do:

21 maja 23:59