

# ZAPDC

## Ćwiczenie 7

### Algorytm RLE

Lev Sergeyev

## 1 Przebieg ćwiczenia

Zaprojektowałem program pozwalający na kompresowanie/dekompresowanie plików binarnych za pomocą algorytmu RLE.

Program pozwala na wybór rozmiaru podstawowego bloku danych, do wyboru 1,2,4 lub 8 bitów (co odpowiada zmiennym typom uint8, uint16, uint32, uint64).

Aby zakodować plik należy uruchomić program z wierszu poleceń z parametrem 'c', 'c1', 'c2', 'c4' lub 'c8' (liczba wskazuje na długość bloku w bajtach) następnie podać nazwę pliku wejściowego i wyjściowego.

Proces kompresowania można rozbić na takie etapy:

- Wczytanie pliku do kompresji.
  - Analiza wczytanych danych:
    - tworzenie słownika bloków,
    - bloków w słowniku w taki sposób, aby najczęściej występujący blok przy kodowaniu był na pierwszym miejscu.
  - Kodowanie danych - przekształcenie na ciąg kodów bloków z liczbą występowania
  - Zapis zakodowanych danych do pliku binarnego
- Skompresowany plik ma następującą strukturę:
- Nagłówek wskazujący długość bloku.
  - Tablica kodów.
  - Ciąg zakodowanej informacji.

Proces kodowania, w zależności od priorytetu bloku i długości ciągu, może zapisać do pliku wyjściowego dane o różnym rozmiarze, np blok, który ma największy priorytet występuje w ciągu 4 takich bloków, w pliku wyjściowym będzie mieć zapis `0b10000100` (na fioletowo - bity kodujące numer bloku, zielone bity kodują długość ciągu).

Innym przykładem może służyć blok, o numerze 5, i długością ciągu 205, ponieważ liczba długości nie może być zapisana w 6ciu bitach, będzie ona kodowana w inny (dłuższy) sposób : `0b00010101,0b11001101` .

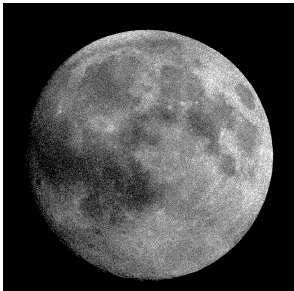
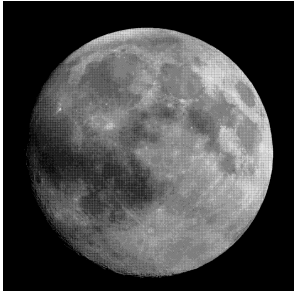

Algorytm pozwala na zapis długości ciągu zarówno jak i kodu bloku o maksymalnej długości 32 bity.

Aby zdekodować plik należy uruchomić program z poleceniem 'd'. Dekodowanie odbywa się w następujący sposób:

- Odczyt nagłówku, który zawiera informacje o rozmiarze bloku.
- Odczyt tablicy kodów.
- Odtworzenie danych do tablicy kodów z długością.
- Odtworzenie pliku oryginalnego.

## 2 Kompresja

Aby sprawdzić działanie algorytmu na obrazach, skonwertowałem pliki w do formatu .bmp przedstawiające bitmapy obrazów. Następnie przeprowadziłem kompresowanie dla każdego trybu. Dodatkowo dołączyłem do porównania plik tekstowy.

Plik	Oryginał	Kompresja c1	Kompresja c2	Kompresja c4	Kompresja c8
	Lysy-losowy.bmp 284.2 KiB	424.0 KiB	342.8 KiB	268.1 KiB	234.9 KiB
	Lysy-wzorcowy.bmp 284.2 KiB	58.8 KiB	60.9 KiB	65.6 KiB	90.0 KiB
	Troll-face-4-RLE.bmp 763.0 KiB	70.2 KiB	83.1 KiB	84.6 KiB	91.3 KiB
Tekst	Lorem_ ipsum.txt 143.4 KiB	215.9 KiB	190.8 KiB	137.9 KiB	127.4 KiB

### 2.1 Kod

<https://github.com/221349/ZAPDC/tree/master/RLE>

## 3 Wnioski

Algorytm Run-Length Encoding jest jednym z najprostrzych algorytmów kompresji bezstratnej.

Najlepiej taki algorytm sprawdza się na prostych rysunkach mających duże obszary tego samego koloru (jak plik "Troll-face-4-RLE.bmp") lub powtarzających się wzorów (jak plik "Lysy-wzorcowy.bmp"). Kompresja plików o zawartości losowej na odwrót może spowodować zwiększenie pliku, jak w przypadku "Lysy-losowy.bmp".

Kompresja tekstu może dać dobry wynik, ale dla większych bloków danych, od 4 znaków.

Wykryte błędy programu: kiedy ilość unikalnych bloków danych jest maksymalna (np 256 dla 'c1'), najczęściej w przypadku plików z danymi losowymi, dekompresja kończy się błędem "seg fault".