Wstęp do programowania w C

Robert Ferens

Lista 7 - 1 grudnia 2021

Zadanie 1

Zapoznaj się z formatem plików graficznych ppm: `https://pl.wikipedia.org/wiki/Portable_anymap` Do listy zostało dołączone zdjęcie w tym formacie(dokładniej P6). Celem programu będzie obróbka zdjęcia zapisanego w tym formacie - wczytanie go ze standardowego wejścia(przekierowanego) i wypisanie na wyjście(przekierowane do pliku). Napisz program, który wczyta ten plik, jego nagłówek wraz z jego binarną zawartością (np. przez (unsigned char)getchar()) i wypisze na wyjście to samo z drobnymi edycjami(może to robić na bieżąco - nie zapisując całego do pamięci):

1. jeśli program zostanie uruchomiony z parametrem "–blackwhite" to powinien pobrać wartości rgb dla każdego piksela, policzyć średnią i zastąpić nią każdą ze składowych - w ten sposób obrazek powinien stać się czarno-biały.
2. jeśli program zostanie uruchomiony z parametrem "–filter" to powinien pobrać drugi parameter składający się z podciągu "rgb". Program powinien tak zmienić plik by pozostały tylko te kolory które znalazły się w parametrze. Np. dla wywołania ./prog -f rb powinien wypisywać normalnie kolor czerwony i niebieski, ale zerować warotści pikseli zielonych.
3. jeśli program zostanie uruchomiony z parametrem "–contrast" to powinien każdą wartość koloru piksela większą od 127 przepisać jako 255, a mniejszą jako 0.
4. jeśli program zostanie uruchomiony z parametrem "–gamma" i drugim paramterem jako wartość gamma (np.0.5) to powinien znormalizować każdą wartość do typu double 0-1, potem podnieść do potęgi gamma i przywrócić do oryginalnego zakresu 0-255 i wypisać. `https://en.wikipedia.org/wiki/Gamma_correction`
5. jeśli program zostanie uruchomiony inaczej poda komunikat o błędzie.

Program może wczytywać dane z przekierowanego wejścia i wypisywać na przekierowane do pliku wyjście, oba typuu ppm. Ewentualnie można użyć operacjach na plikach.

UWAGA! Linux powinien normalnie wyświetlać pliki ppm, Windowsowi pewnie będzie potrzebny program graficzny by je zobaczyć np. gimp, krita, adobe - pozostawiam to do sprawdzenia. W ostateczności istnieją internetowe konwertery ppm do png,jpg itd.

Zadanie 2

Na końcu ostatniego wykładu wprowadzona została struktura stosu. Stos jest strukturą pozwalającą nam wkładać na niego elemtenty i wyciągać w przeciwnej kolejności do tej w jakiej zostały włożone, tzn. ostatni włożony element jest wyciągany jako pierwszy. Został on na wykładzie zaimplementowany na podstawie listy. Lista jest strkturą, której każdy element wskazuje na następny. Zaletą tej struktury jest łatwość włożenia i usunięcia elementu w dowolnym jej miejscu niskim kosztem (zastanów się jak trudne byłoby to w tablicy - wymagało by przesunięcia całej tablicy po prawej stronie by coś dołożyć - podobnie z usuwaniem). Wadą natomiast brak szybkiego dostępu do konkretnego elementu - do k-tego elemtentu trzeba przejść przed k-iteracji.

Zaznajom się z implementacją stosu z wykładu i dodaj do implementacji trzy funkcje, by stał się pełnoprawną listą - pierwsza insert(p,x) powinna dodawać x wartość na p-tej pozycji na liście, druga del(p) usuwać p-ty element na liście, trzecia get(p) powinna zwrócić wartość p-tego elementu. Napisz także funkcję $list\_print(structe\_stos^{*}lista)$ która wypisuje wszystkie elementy na liście(stosie) oddzielone spacjami. Pomoże ona debugować program. Pamiętaj, że wpisywany tekst potrzebuje swojego trwałego miejsca w pamięci.

Punktacja:

1. insert - 3pkt
2. del - 3pkt
3. get - 2pkt
4. print -2 pkt

Wszystko powinno zostać przetestowane przez następujące operacje(po każdej wypisanie listy):

```
włóż na początek "mi"
włóż na początek "ko"
włóż na początek "la"
włóż na początek "jki"
usuń 3-ci element
usuń 1-wszy element
dadaj po 2-gim elemencie "do"
dodaj po pierwszym elemencie "re"
```

Spodziewany wynik:

```
mi
ko mi
la ko mi
jki la ko mi
jki la mi
la mi
la mi do
la re mi do
```

Zadanie 3

Dostępne ze sprawdzaczką na skosie.