Tema 3. Gigel si captcha-urile

Responsabili:

- Birsu Ion

Termen de predare: **15.01.2018**

Termen de predare hard: **21.01.2018**

Actualizări:

14/12/17 - adaugare bonus si checker bonus;

14/12/17 - adaugare vmchecker;

02/01/18 - au fost reparate testele de la bonus;

08/01/18 - update teste;

14/01/18 - modificare deadline

Scopul temei:

- utilizarea structurilor de date;
- lucrul cu fisiere, atat text, cat si binare;
- abordarea cu pasi mici a unei probleme mai complexe;

Introducere

Resurse generale

- Regulament: seria CA
- Regulament: seria CB/CD
- Punctaj seria CA
- Calendar
- Catalog laborator
- Debugging
- Coding style
- Checker laborator CB/CD

Cursuri

Continutul Tematic

Laboratoare

01. Unelte de programare
02. Tipuri de date. Operatori.
03. Instrucțiunile limbajului C
04. Funcții
05. Tablouri. Particularizare - vectori
06. Matrice. Operații cu matrice

Gigel a descoperit un site pe care, pentru fiecare cont creat, va fi recompensat cu o suma de bani. Primul lucru la care s-a gandit Gigel a fost sa scrie un script ce genereaza automat utilizatori. Totusi, mai are o problema majora de rezolvat: pentru fiecare cont creat trebuie rezolvat cate un captcha.

Desi aparent rezolvarea unui captcha pare foarte greu de automatizat, Gigel a observat ca toate imaginile pe care le primeste sunt in format bmp si sunt formate doar din cifre de dimensiuni egale si scrise mereu in acelasi mod. Cu toate acestea, nu se simte in stare sa rezolve de unul singur aceasta problema si apeleaza din nou la ajutorul vostru. O sa fiti rasplatiti cu inca 100 de puncte daca il ajutati si de aceasta data.

Structura formatului BMP

Vom lucra cu fisiere **BMP**, deci, cu **fisiere binare**.

O imagine BMP are urmatoarea structura:

- un **File Header** care are urmatoarele campuri:
  1. **signature** – 2 octeti - literele 'B' si 'M' in ASCII;
  2. **file size** – 4 octeti – dimensiunea intregului fisier;
  3. **reserved** – 4 octeti – nefolosit;
  4. **offset** – 4 octeti – offsetul de la inceputul fisierului pana la inceputului bitmap-ului, adica al matricii de pixeli.
- un **Info Header** care poate avea structuri diferite, insa noi vom lucra cu cel care se numeste **BITMAPINFOHEADER**. Are urmatoarele campuri:
  1. **size** – 4 octeti – dimensiunea Info Header-ului, care are o valoare fixa, 40;
  2. **width** – 4 octeti – latimea matricii de pixeli (numarul de coloane);
  3. **height** – 4 octeti – inaltimea matricii de pixeli (numarul de randuri);
  4. **planes** – 2 octeti – setat la valoarea fixa 1;
  5. **bit count** – 2 octeti – numarul de biti per pixel. In cazul nostru va avea mereu valoarea 24, adica reprezentam fiecare pixel pe 3 octeti, adica cele 3 canale, RGB;
  6. **compression** – 4 octeti – tipul de compresie. Acest camp va fi 0;
  7. **image size** – 4 octeti – se refera la dimensiunea matricii de pixeli, inclusiv padding-ul adaugat (vedeti mai jos);

8. **x pixels per meter** – 4 octeti – se refera la rezolutia de printare. Pentru a simplifica putin tema, veti seta acest camp pe 0. Nu o sa printam imaginile :).
9. **y pixels per meter** – la fel ca mai sus;
10. **colors used** – numarul de culori din paleta de culori. Aceasta este o sectiune care va lipsi din imaginile noastre BMP, deoarece ea se afla in general imediat dupa **Info Header** insa doar pentru imaginile care au campul **bit count** mai mic sau egal cu 8. Prin urmare, campul va fi setat pe 0;
11. **colors important** – numarul de culori importante. Va fi, de asemenea, setat pe 0, ceea ce inseamna ca toate culorile sunt importante.

- **BitMap**-ul, care este matricea de pixeli si care ocupa cea mai mare parte din fisier. Trei lucruri trebuie mentionate despre aceasta:
  1. pixelii propriu-zisi se afla intr-o matrice de dimensiune **height x width**, insa ea poate avea o dimensiune mai mare de atat din cauza **paddingului**. Acest padding este adaugat la sfarsitul fiecarei linii astfel incat fiecare linie sa inceapa de la o adresa (offset fata de inceputul fisierului) multiplu de 4. Mare atentie la citire, pentru ca acest padding trebuie **ignorat** (fseek). De asemenea, la scriere va trebui sa puneti **explicit** valoarea 0 pe toti octetii de padding.
  2. este **rasturnata**, ceea ce inseamna ca prima linie in matrice contine de fapt pixelii din extremitatea de jos a imaginii. Vedeti exemplul de mai jos;
  3. canelele pentru fiecare pixel sunt in ordinea BGR (**B**lue **G**reen **R**ed).

Header-ele pe care le puteti folosi in implementare se afla in scheletul de cod asociat temei.

Urmatiti cu foarte mare atentie exemplul de aici si incercati sa intelegeti cum e reprezentata o imagine BMP **inainte** de a incepe implementarea. Daca e ceva neclar, puteti intreba oricand pe forum.

Imaginea pe care o primiti are fundalul perfect alb (255, 255, 255) si contine doar cifre scrise cu diferite nuante de culori. Cifrele au dimensiunea de 5×5 pixeli si nu se

pot intersecta. Daca notam cu "X" un pixel colorat si cu "." un pixel alb cifrele arata in felul urmator:

```
XXXXX   ....X   XXXXX   XXXXX   X...X   XXXXX   XXXXX   XXXXX   XXXXX
X...X   ....X   ....X   ....X   X...X   X....   X....   ....X   X...X
X...X   ....X   XXXXX   XXXXX   XXXXX   XXXXX   XXXXX   ....X   XXXXX
X...X   ....X   X....   ....X   ....X   ....X   X...X   ....X   X...X
XXXXX   ....X   XXXXX   XXXXX   ....X   XXXXX   XXXXX   ....X   XXXXX
```

Cerinte

Toate taskurile (mai putin bonusul) vor procesa o singura imagine. Numele acesteia o sa fie citit de pe prima linie din fisierul de intrare (vezi formatul datelor de intrare). Scopul fiecarui task este crearea unui unui nou fisier (binar pentru taskurile 1 si 3 si text pentru taskul 2) dupa urmatoarele reguli:

- se elimina extensia ".bmp" din fisierul initial.
- se adauga:
  - **"_task1.bmp"** pentru primul task;
  - **"_task2.txt"** pentru cel de-al doilea task;
  - **"_task3.bmp"** pentru taskul trei.

Exemplu: daca imaginea initiala se numeste **"captcha.bmp"**, rezolvarea completa a celor trei taskuri presupune existenta fisierelor **"captcha_task1.bmp"**, **"captcha_task2.txt"** si **"captcha_task3.bmp"** dupa rularea programului.

Bonusul va procesa o imagine diferita (citita din acelasi fisier de intrare) si presupune formarea unui noi imagini dupa reguli asemanatoare primelor taskuri, adaugand insa **"_bonus.bmp"** la final de nume. (imaginea **"captchaB.bmp"** va crea fisierul text **"captchaB_bonus.bmp"**).

Task1 (20p)

Inainte de a incerca rezolvarea captcha-ului, Gigel va propune sa incepeti cu ceva simplu: schimbarea culorii tuturor cifrelor din imagine. Pe langa imaginea cu care o sa lucrati la primele 3 taskuri o sa mai primiti 3 numere intregi intre 0 si 255 (in formatul BGR) reprezentand noua culoare dorita a cifrelor. Rezolvarea corecta a taskului presupune existenta unei imagini noi cu aceleasi cifre, dar colorate dupa regula data.

Exemplu input:

(captcha.bmp)

+ valorile 0, 255, 0 (B, G, R)

Exemplu output:

(captcha_task1.bmp)

Observatii:

- cu exceptia matricei de pixeli, toate celelalte valori din imaginea initiala (valorile campurilor din headere) trebuie sa ramana neschimbate;

Task2 (30p):

Acum, dupa ce ati reusit (sau nu) sa colorati cifrele din captcha a venit momentul sa incercati si recunoasterea lor. Numarul format va avea cifrele ordonate de la stanga la dreapta.

Exemplu input:

(captcha.bmp)

Exemplu output:

26908109 **(captcha_task2.txt)**

Observatii:

- cifrele pot incepe oriunde in imagine (atata timp cat nu depasesc dimensiunea imaginii).
- cifrele nu se vor intersecta si oricare doua cifre vor fi separare de cel putin o linie sau o coloana;
- nu vor exista 2 cifre care sa inceapa pe aceeasi coloana, putand astfel sa spunem pentru oricare 2 cifre care se afla la stanga celeilalte;
- pentru simplitate, putem considera ca fiecare cifra reprezinta un patrat de pixeli de dimensiune 5 * 5. Deci, nu va aparea urmatorul caz in fisierele de test:

```
XXXXX...X.
X...X...X.
X...X...X.
X...X...X.
XXXXX...X.
```

- Testul de mai sus nu e valid din cauza cifrei unu. Aceasta trebuie reprezentata ca o matrice de 5 * 5 pixeli in care primele 4 coloane sa fie albe (lucru care nu se intampla).

Task3 (30p):

Vazand ca lucrurile merg bine si este pe cale sa devina milionar, Gigel mai are o rugaminte la voi. Doreste sa faca o statistica a cifrelor care apar si sa vada cum ar arata captcha-urile fara anumite cifre. O sa primiti o lista de cifre pe care va trebui sa le eliminati. Dupa eliminarea unei cifre toate celelalte cifrele ce se afla la dreapta ei vor fi mutate spre stanga in felul urmator:

- prima cifra de dupa o cifra eliminata va lua locul cifrei eliminate;

- urmatoarea cifra va lua locul cifrei recent mutate;
- se va repeta pasul 2 pana nu mai raman cifre;

Exemplu input:

(captcha.bmp)

+ vectorul de valori: "2 0 9"

Exemplu output:

(captcha_task3.bmp)

Observatii:

- dimensiunea imaginii precum si celelalte campuri din headerele imaginii originale vor ramane neschimbate si in cadrul acestui task;

BONUS (20p):

Dupa ce Gigel a reusit sa treaca de captcha si scoata o suma mare de bani, s-a gandit sa le lase o mica surpriza administratorilor site-ului. Astfel, pentru ca el iubeste luminitele de craciun, s-a gandit sa adauge cateva puncte random pe imagine. Aceste puncte vor avea marime fixa de un pixel si raza de actiune 7. Punctele reprezinta beculete de craciun, beculete ce vor lumina cifrele din captcha-ul nostru. Culoarea fiecarui pixel ce apartine unei cifre se va schimba in functie de culoarea initiala si culorile beculetelor in a caror raza se afla. Pentru calcularea noii culori se va folosi o medie aritmetica asupra fiecare valori din RGB.

Gigel va roaga sa generati imaginea dupa ce ati colorat cifrele in functie de luminite.

Exemplu input:

 **(captchaB.bmp)**

Exemplu output:

 **(captchaB_bonus.bmp)**

Formatul datelor de intrare:

Datele de intrare se vor citi din fisierul **"input.txt"**. Acesta va contine 4 linii, prima dintre ele avand numele imaginii folosite la primele 3 taskuri, cea de-a doua va contine 3 numere intregi ce vor fi folosite pentru colorarea ceruta la primul task, a 3a linie contine numerele ce trebuie eliminate la taskul3, iar ultima linie reprezinta numele imaginii folosite pentru bonus.

Exemplu:

```
captcha.bmp             // fisierul de procesat la primele 3 taskuri
0 255 0                 // culoarea ce va fi folosita la primul task (B
0 7 9                   // cifrele ce trebuie eliminate din captcha in
captchaB.bmp            // imaginea folosita la bonus
```

Formatul datelor de iesire:

Pentru fiecare task rezolvat va trebui creat cate un fisier (cu extensie .txt sau .bmp) pe baza carora o sa se verifice corectitudinea temei.

Observatii si mentiuni:

- task-urile au un total de **80 puncte** (fara bonus). Alte 20 de puncte se vor acorda pentru coding style si README.

- **in cadrul fiecarui task, primele (i * 10)% din teste vor fi alcatuite doar din cifre cuprinse intre 0 si min(9, i) (1 ≤ i ≤ 10)**. Puteti incepe cu o rezolvare simplista (care sa detecteze doar 0 si 1) si sa continuati cu celelalte cifre doar dupa ce primiti primele puncte;
- **rezolvarea oricarui task nu este conditionata de rezolvarea/corectitudinea celorlalte**. Daca vreti sa rezolvati doar taskurile 3 si bonusul faceti doar fisierele specifice lor si o sa primiti punctajul corespunzator.
- nicio imagine nu va avea latimea sau inaltimea mai mare de 100;
- deoarece imaginile au rezolutie mica, acestea pot fi vizualizate mai bine folosind un program de manipulare a imaginilor (de exemplu GIMP)
- tema va fi incarcata pe vmchecker si va fi corectata cu testerul local pus la dispozitie in curand;

Arhiva pentru vmchecker va contine:

- Makefile, cu targeturi de build (pentru a genera fisierul/fisierele executabile necesare), run (pentru a rula executabilul/executabilele generate) si clean (care va sterge toate executabilele si fisierele binare/text facute de voi).
- sursele voastre, adica fisierele .c si .h. Inclusiv headerul bmp_header.h sau sub orice alta denumire il folositi;
- README, in care trebuie sa dati detalii despre implementare, de ce ati ales sa rezolvati intr-un anumit fel, etc.

Resurse:

Resursele pentru tema se pot descarca de aici. Sunt prezente:

- **bmp_header.h**: headerul care contine declaratiile struct-urilor pe care le veti folosi in citirea unui fisier BMP;
- **checkerul** si **testele** folosite pentru corectarea temei;

Imaginile din enuntul temei nu au formatul BMP (nu este permisa incarcarea fisierelor BMP pe aceasta pagina). Folositi doar imaginile din arhiva de mai sus pentru a va testa tema!

programare/teme_2017/tema3_2017_cbd.txt · Last modified: 2018/01/16 00:49 by ion.birsu

Old revisions Media Manager Back to top