4η Εργασία – Κρυφό μήνυμα σε εικόνα

Ανάλυση του προβλήματος

Το πρόγραμμα απόκρυψης πρέπει αρχικά να ζητάει από τον χρήστη να πληκτρολογήσει τόσο το όνομα του αρχείου κειμένου όσο και το όνομα του αρχείου εικόνας μέσα στο οποίο θα κρυφτεί το κείμενο. Στη συνέχεια, το πρόγραμμα θα διαβάζει την κεφαλίδα του αρχείου εικόνας και θα ελέγχει αν το συγκεκριμένο αρχείο έχει τη δυνατότητα απόκρυψης κειμένου. Για να έχει αυτή τη δυνατότητα, πρέπει να είναι τύπου bmp, με βάθος χρώματος 24 bit και με τις κατάλληλες διαστάσεις ώστε να διαθέτει byte συμπλήρωσης. Ανάλογα με τις διαστάσεις της εικόνας, υπολογίζουμε τα διαθέσιμα byte συμπλήρωσης και τοποθετούμε τον δείκτη εγγραφής στην αρχή του πρώτου από αυτά τα byte. Στη συνέχεια γράφουμε στη θέση αυτή τόσους χαρακτήρες όσο είναι το πλήθος των byte συμπλήρωσης, τους οποίους διαβάζουμε από το αρχείο κειμένου. Κατόπιν, τοποθετούμε τον δείκτη εγγραφής στο επόμενο διαθέσιμο byte συμπλήρωσης, που βρίσκεται στο τέλος της επόμενης σειράς pixel. Τα παραπάνω δύο βήματα επαναλαμβάνονται μέχρι να διαβάσουμε όλους τους χαρακτήρες από το αρχείο κειμένου ή μέχρι να εξαντληθούν τα διαθέσιμα byte συμπλήρωσης που βρίσκονται στην εικόνα.

Το πρόγραμμα που θα εξάγει ένα κρυμμένο κείμενο μέσα από μια εικόνα θα λειτουργεί με παρόμοιο τρόπο. Αρχικά ζητάει από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου εικόνας η οποία περιέχει το κρυμμένο κείμενο, και έπειτα διαβάζει την κεφαλίδα του αρχείου εικόνας για να ελέγξει αν το συγκεκριμένο αρχείο έχει τη δυνατότητα απόκρυψης κειμένου. Αν δεν την έχει (δηλαδή, δεν είναι τύπου bmp με βάθος χρώματος 24 bit ή δεν διαθέτει byte συμπλήρωσης), εμφανίζει το κατάλληλο μήνυμα και η διαδικασία διακόπτεται. Ακολούθως, τοποθετεί τον δείκτη ανάγνωσης στην αρχή του πρώτου byte συμπλήρωσης και διαβάζει από τη θέση αυτή τόσους χαρακτήρες όσο είναι το πλήθος των byte συμπλήρωσης, τους οποίους γράφει σε ένα αρχείο εξόδου. Στη συνέχεια μετακινούμε τον δείκτη ανάγνωσης στο επόμενο διαθέσιμο byte συμπλήρωσης και διαβάζουμε τους επόμενους κρυμμένους χαρακτήρες. Η διαδικασία αυτή επαναλαμβάνεται μέχρι να διαβάσουμε όλους τους κρυμμένους χαρακτήρες, δηλαδή μέχρι να εντοπίσουμε byte συμπλήρωσης με μηδενική τιμή ή μέχρι να εξαντληθούν όλα τα διαθέσιμα byte συμπλήρωσης της εικόνας.

Η υλοποίηση της εργασίας βασίζεται στην ιδιαίτερη δομή των αρχείων εικόνων τύπου BMP η οποία αναλύεται στα παραδείγματα Π14.8 και Π14.9. Ο κώδικας που χρησιμοποιείται είναι μια προσαρμογή του κώδικα των παραπάνω παραδειγμάτων.

Βήματα

❶ Το πρόγραμμα απόκρυψης πρέπει αρχικά να ζητάει από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου εικόνας μέσα στο οποίο θα κρυφτεί το κείμενο. Στη συνέχεια το πρόγραμμα διαβάζει την κεφαλίδα του αρχείου εικόνας και εμφανίζει κάποια από τα στοιχεία αυτά για να ενημερώσει τον χρήστη.

❷ Μετά, το πρόγραμμα υπολογίζει το πλήθος των byte συμπλήρωσης (padding) που υπάρχουν σε κάθε γραμμή pixel της εικόνας, και στη συνέχεια ελέγχει αν το αρχείο είναι κατάλληλο για την απόκρυψη κειμένου. Συγκεκριμένα ελέγχει αν είναι τύπου bmp με βάθος χρώματος 24 bit και αν διαθέτει byte συμπλήρωσης. Αν το αρχείο είναι κατάλληλο, υπολογίζει και εμφανίζει το πλήθος των χαρακτήρων που μπορούμε να κρύψουμε μέσα στο αρχείο, διαφορετικά η διαδικασία σταματά και εμφανίζεται ένα κατάλληλο μήνυμα.

❸ Στη συνέχεια ζητείται από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου το οποίο περιέχει το κείμενο που θέλουμε να κρύψουμε μέσα στην εικόνα.

❹ Με βάση την απόσταση (offset) στην οποία βρίσκονται τα δεδομένα των pixel της εικόνας, το πλάτος της εικόνας σε pixel (platos) και το πλήθος των byte συμπλήρωσης (padding), υπολογίζουμε τη θέση

στην οποία βρίσκεται το πρώτο byte συμπλήρωσης κάθε γραμμής (g). Στη συνέχεια ο δείκτης εγγραφής τοποθετείται στη θέση του πρώτου byte συμπλήρωσης της γραμμής g.

❺ Μετά το πρόγραμμα διαβάζει από το αρχείο κειμένου έναν χαρακτήρα και τον γράφει μέσα στο αρχείο εικόνας στη θέση ενός byte συμπλήρωσης. Αυτό επαναλαμβάνεται τόσες φορές όσα είναι τα byte συμπλήρωσης κάθε γραμμής. Τα βήματα 4 και 5 επαναλαμβάνονται για κάθε γραμμή pixel της εικόνας, μέχρι να εξαντληθούν οι χαρακτήρες του αρχείου κειμένου ή τα byte συμπλήρωσης που υπάρχουν στην εικόνα.

❻ Τέλος, εμφανίζει το πλήθος χαρακτήρων που έχουν κρυφτεί μέσα στην εικόνα. Στην περίπτωση που τα byte συμπλήρωσης του αρχείου εικόνας δεν επαρκούν για την απόκρυψη του συνόλου των χαρακτήρων, τότε οι χαρακτήρες ενδέχεται να είναι λιγότεροι από τους χαρακτήρες του αρχείου κειμένου.

Κώδικας – Απόκρυψη κειμένου μέσα σε εικόνα

e4_a.c

```
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
   FILE *fp1,*fp2;
   char set;
   char onoma_eikonas[30],onoma_keimenoy[30];
   int platos,ypsos,offset,g,size,bytes_per_raw,padding,padding_pos;
   int s,count=0,i,cnt;
   short int bits;
   unsigned char ch;
   printf("Δώσε όνομα εικόνας στην οποία θα κρύψω το κείμενο: ");
   gets(onoma_eikonas);
   fp1=fopen(onoma_eikonas,"rb+");
   if (fp1==NULL)
   {
      printf("Το αρχείο εικόνας %s δεν υπάρχει\n",onoma_eikonas);
      exit(1);
   }
   fseek(fp1,2,SEEK_SET);
   fread(&size,4,1,fp1);
   printf("Byte:%d\n",size);
   fseek(fp1,10,SEEK_SET);
   fread(&offset,4,1,fp1);
   printf("Απόσταση δεδομένων pixel:%d\n",offset);
   fseek(fp1,18,SEEK_SET);
   fread(&platos,4,1,fp1);
   fread(&ypsos,4,1,fp1);
   printf("Διαστάσεις εικόνας:%dx%d\n",platos,ypsos);
   bytes_per_raw=3*platos;
   if (bytes_per_raw%4==0)
      padding=0;
   else
      padding=4-bytes_per_raw%4;
   printf("Byte συμπλήρωσης:%d\n",padding);
```

❶ Ζητείται από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου εικόνας μέσα στο οποίο θα κρύψουμε το κείμενο.

❶ Διαβάζουμε στοιχεία από την κεφαλίδα του αρχείου εικόνας

❷ Υπολογίζει και εμφανίζει τα διαθέσιμα byte συμπλήρωσης που υπάρχουν σε κάθε γραμμή pixel της εικόνας.

```
if (padding==0)
{
   printf("Συγγνώμη αλλά το πρόγραμμα δουλεύει μόνο με εικόνες που έχουν byte
           συμπλήρωσης\n");
   printf("Δοκίμασε με μια εικόνα bmp 999x999 pixel\n");
   exit(1);
}
fseek(fp1,28,SEEK_SET);
fread(&bits,2,1,fp1);
printf("Bit ανά pixel:%d\n",bits);
if (bits!=24)
{
   printf("Συγγνώμη αλλά το αρχείο δεν είναι RGB 24bit\n");
   exit(1);
}
printf("\n--> Στο αρχείο %s μπορούν να κρυφτούν μέχρι%d χαρακτήρες!\n\n",
        onoma_eikonas,ypsos*padding);
printf("Δώσε όνομα αρχείου κειμένου για απόκρυψη: ");
gets(onoma_keimenoy);
fp2=fopen(onoma_keimenoy,"rb");
if (fp2==NULL)
{
   printf("Το αρχείο κειμένου %s δεν υπάρχει\n",onoma_keimenoy);
   exit(1);
}
printf("\n===== Πάτησε <ENTER> για συνέχεια =====\n");
getchar();
for (g=0;g<ypsos;g++)
{
   padding_pos=offset+platos*3*(g+1)+padding*g;
   fseek(fp1,padding_pos,SEEK_SET);
   for (i=1;i<=padding;i++)
   {
      if (fread(&set,1,1,fp2))
      {
         fwrite(&set,1,1,fp1);
         putchar('#');
         count++;
      }
      else
      {
         set=0;
         fwrite(&set,1,1,fp1);
         continue;
      }
   }
   if (feof(fp2)) break;
}
```

❷ Στην περίπτωση που το αρχείο δεν διαθέτει byte συμπλήρωσης ή δεν έχει βάθος χρώματος 24 bit, η διαδικασία σταματά και εμφανίζεται το ανάλογο μήνυμα.

❷ Ανάλογα με τις διαστάσεις της εικόνας και τα byte συμπλήρωσης ανά γραμμή, υπολογίζει και εμφανίζει το πλήθος των χαρακτήρων που μπορούμε να κρύψουμε σε αυτό το αρχείο.

❸ Ζητείται από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου κειμένου.

❹ Υπολογίζει τη θέση του πρώτου byte συμπλήρωσης της γραμμής **g** της εικόνας..

❹ Τοποθετεί τον δείκτη ανάγνωσης στη θέση του πρώτου byte συμπλήρωσης της κάθε γραμμής.

❺ Διαβάζει από το αρχείο κειμένου (fp2) έναν χαρακτήρα και τον γράφει μέσα στο αρχείο εικόνας (fp1) στη θέση ενός byte συμπλήρωσης. Αυτό γίνεται τόσες φορές όσα είναι τα byte συμπλήρωσης κάθε γραμμής.

❺ Αν έχουμε φτάσει στο τέλος του αρχείου κειμένου, συνεχίζει να γράφει μηδενικά byte συμπλήρωσης, για τη συγκεκριμένη γραμμή του αρχείου εικόνας,.

```
   fclose(fp1);
   fclose(fp2);
   printf("\nΣτην εικόνα %s κρύφτηκαν %d χαρακτήρες\n",onoma_eikonas, count);
   return 0;
}
```

❻ Εμφανίζει το πλήθος των χαρακτήρων που έχουν κρυφτεί μέσα στην εικόνα.

Κώδικας – Ανάκτηση κρυμμένου κειμένου από εικόνα

Για να ανακτήσουμε ένα κρυμμένο κείμενο που βρίσκεται μέσα σε μια εικόνα, ακολουθούμε τα παρακάτω βήματα:

Βήματα

❶ Το πρόγραμμα ανάκτησης του κειμένου αρχικά να ζητάει από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου εικόνας η οποία περιέχει το κρυμμένο κείμενο. Στη συνέχεια, το πρόγραμμα διαβάζει την κεφαλίδα του αρχείου εικόνας και εμφανίζει κάποια από τα στοιχεία αυτά για να ενημερώσει τον χρήστη.

❷ Έπειτα υπολογίζει το πλήθος των byte συμπλήρωσης (padding) που υπάρχουν σε κάθε γραμμή pixel της εικόνας και στη συνέχεια ελέγχει αν το αρχείο είναι κατάλληλο για την απόκρυψη κειμένου. Συγκεκριμένα, ελέγχει αν είναι τύπου bmp με βάθος χρώματος 24 bit και αν διαθέτει byte συμπλήρωσης. Αν το αρχείο δεν είναι κατάλληλο, οπότε δεν είναι δυνατόν να περιέχει κρυμμένο κείμενο, η διαδικασία σταματά και εμφανίζεται ένα κατάλληλο μήνυμα.

❸ Κατόπιν, υπολογίζει τη θέση στην οποία βρίσκεται το πρώτο byte συμπλήρωσης κάθε γραμμής (g) και τοποθετεί εκεί τον δείκτη ανάγνωσης.

❹ Μετά, το πρόγραμμα διαβάζει τα byte συμπλήρωσης, τα οποία περιέχουν τους κρυμμένους χαρακτήρες, και τους γράφει στο αρχείο εξόδου. Αυτό γίνεται τόσες φορές όσα είναι τα byte συμπλήρωσης κάθε γραμμής. Τα βήματα 3 και 4 επαναλαμβάνονται για κάθε γραμμή pixel της εικόνας και σταματά όταν διαβάσει μηδενικό byte συμπλήρωσης ή όταν εξαντληθούν οι γραμμές των pixel της εικόνας.

❺ Τέλος, εμφανίζει το πλήθος χαρακτήρων που διάβασε καθώς και το όνομα του αρχείου εξόδου στο οποίο έχουν γραφεί.

e4_b.c

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   FILE *fp1,*fp3;
   char set;
   char onoma_eikonas[30],onoma_keimenoy[30];
   int platos,ypsos,offset,g,size,bytes_per_raw,padding,padding_pos;
   int s,count=0,i,cnt;
   short int bits;
   unsigned char  ch;
   printf("Δώσε όνομα αρχείου εικόνας στην οποία είναι κρυμμένο το κείμενο: ");
   gets(onoma_eikonas);
   fp1=fopen(onoma_eikonas,"rb");
   if (fp1==NULL)
   {
      printf("Το αρχείο εικόνας %s δεν υπάρχει\n",onoma_eikonas);
      exit(1);
   }
```

❶ Ζητείται από τον χρήστη να πληκτρολογήσει το όνομα του αρχείου εικόνας μέσα στην οποίο υπάρχει κρυμμένο κείμενο.

```
fp3=fopen("out.txt","w");
fseek(fp1,2,SEEK_SET);
fread(&size,4,1,fp1);
printf("Byte:%d\n",size);
fseek(fp1,10,SEEK_SET);
fread(&offset,4,1,fp1);
printf("Απόσταση δεδομένων pixel:%d\n",offset);
fseek(fp1,18,SEEK_SET);
fread(&platos,4,1,fp1);
fread(&ypsos,4,1,fp1);
printf("Διαστάσεις εικόνας:%dx%d\n",platos,ypsos);
bytes_per_raw=3*platos;
if (bytes_per_raw%4==0)
   padding=0;
else
   padding=4-bytes_per_raw%4;
printf("Byte συμπλήρωσης:%d\n",padding);
if (padding==0)
{
   printf("Συγγνώμη αλλά το αρχείο εικόνας δεν διαθέτει byte συμπλήρωσης\n");
   printf("οπότε δεν είναι δυνατόν να περιέχει κρυμμένο κείμενο\n");
   exit(1);
}
fseek(fp1,28,SEEK_SET);
fread(&bits,2,1,fp1);
printf("Bit ανά pixel:%d\n",bits);
if (bits!=24)
{
   printf("Συγγνώμη αλλά το αρχείο δεν είναι RGB 24bit\n");
   exit(1);
}
printf("===== Πάτησε <ENTER> για συνέχεια =====\n");
getchar();
for (g=0;g<ypsos;g++)
{
   padding_pos=offset+platos*3*(g+1)+padding*g;
   fseek(fp1,padding_pos,SEEK_SET);
   for (i=1;i<=padding;i++)
   {
      fread(&set,1,1,fp1);
      count++;
      putchar('#');
      if (set!=0)
         fprintf(fp3,"%c",set);
      else
         break;
   }
   if (set==0)
```

❶ Διαβάζει και εμφανίζει στοιχεία από την κεφαλίδα του αρχείου εικόνας

❷ Υπολογίζει και εμφανίζει τα διαθέσιμα byte συμπλήρωσης που υπάρχουν σε κάθε γραμμή pixel της εικόνας.

❷ Στην περίπτωση που το αρχείο δεν διαθέτει byte συμπλήρωσης ή δεν έχει βάθος χρώματος 24 bit, η διαδικασία σταματά και εμφανίζεται το ανάλογο μήνυμα.

❸ Υπολογίζει τη θέση του πρώτου byte συμπλήρωσης της γραμμής **g** της εικόνας.

❸ Τοποθετεί τον δείκτη ανάγνωσης στη θέση του πρώτου byte συμπλήρωσης της κάθε γραμμής.

❹ Διαβάζει από το αρχείο εικόνας (fp1), και συγκεκριμένα από τη θέση ενός byte συμπλήρωσης, έναν χαρακτήρα και τον γράφει στο αρχείο εξόδου (fp3). Αυτό γίνεται τόσες φορές όσα είναι τα byte συμπλήρωσης κάθε γραμμής. Σταματά όταν διαβάσει μηδενικό byte συμπλήρωσης ή όταν εξαντληθούν οι γραμμές των pixel της εικόνας.

```
      {
         printf("\n\Διαβάστηκαν %d κρυμμένοι χαρακτήρες!",count);
         printf("\nΤο κρυμμένο κείμενο βρίσκεται στο αρχείο out.txt\n");
         break;
      }
   }
   fclose(fp1);
   fclose(fp3);
   return 0;
}
```

❺ Εμφανίζει το πλήθος των κρυμμένων χαρακτήρων που έχουν εξαχθεί από το αρχείο εικόνας, καθώς και το όνομα του αρχείου στο οποίο έχουν γραφεί.

Προτάσεις

Η εργασία αυτή επιδέχεται πάρα πολλές τροποποιήσεις ανάλογα με τη δομή των αρχείων εικόνων. Μια ιδέα θα ήταν το κείμενο να μην κρύβεται μέσα στα byte συμπλήρωσης αλλά μέσα στα δεδομένα των pixel της εικόνας. Χρησιμοποιώντας την τεχνική του παραδείγματος Π14.9, θα μπορούσαμε να κρύψουμε μέσα σε κάθε pixel μιας εικόνας με βάθος χρώματος 24 bit τρεις χαρακτήρες. Αυτό θα έδινε τη δυνατότητα απόκρυψης πολύ περισσότερων χαρακτήρων. Για παράδειγμα, σε μια εικόνα διαστάσεων 1000x1000 pixel θα μπορούσαν να κρυφτούν μέχρι 3.000.000 χαρακτήρες!

Η τεχνική που προτείνεται είναι να εκτελείται η πράξη XOR (^) των byte που συνθέτουν το κάθε pixel με τους χαρακτήρες που θέλουμε να κρύψουμε, ένα προς έναν με τη σειρά. Φυσικά η εικόνα θα επηρεαζόταν. Ο παραλήπτης έπειτα θα λάμβανε δύο εικόνες, την αρχική και αυτή που προέκυψε μετά από την απόκρυψη των χαρακτήρων. Έπειτα θα εφαρμοζόταν πάλι η πράξη XOR (^) μεταξύ των byte των αντίστοιχων pixel των δύο εικόνων και θα εξάγονταν οι κρυμμένοι χαρακτήρες!