Άσκηση 13

Δομή Δεδομένων

- Για κάθε καταθέτη: *… Κάθε καταγραφή περιλαμβάνει το επώνυμο του κατόχου του, λογαριασμού, το ύψος του υπολοίπου, καθώς και το τρέχον επιτόκιο, και την ημερομηνία που "άνοιξε", ο λογαριασμός, με την ακόλουθη μορφή*:
  - Nikolaidis, 20000, 3.5, 2006

Δομή Δεδομένων

```
typedef struct {
    char owner[40];
    long balance;
    int year;
    float rate;
} accountT;
```

Πολλοί Καταθέτες

- Αρχείο "bank_new.dat"
  - Πόσοι είναι και τα στοιχεία τους.
- Αρχείο "bank_old.dat"
  - Πόσοι είναι και τα στοιχεία τους.
- Καταθέτες στο αρχείο bank_new και όχι στο bank_old.
  - Πόσοι είναι και τα στοιχεία τους.

Πολλοί Καταθέτες

- Αρχείο "bank_new.dat"
  - accountT newData[100]; int newDataSize;
- Αρχείο "bank_old.dat"
  - accountT oldData[100]; int oldDataSize;
- Καταθέτες στο αρχείο bank_new και όχι στο bank_old.
  - accountT missing[100]; int mSize;

Ανάγνωση Στοιχείων Αρχείων

- Συνάρτηση readFileData
  - Παράμετροι?
  - Τύπος?

```
??? readFileData(???,  ??? );
```

Ανάγνωση Στοιχείων Αρχείων

- Συνάρτηση readFileData
  - Παράμετροι?
  - Τύπος?

```
int readFileData(char fileName[30],  accountT tab[]);
```

```
int readFileData(char fileName[30], accountT tab[]) {
   FILE* infile;
   int count = 0; int nscan;
   char termch;
   infile = fopen(fileName, "r");
   if (infile == NULL)  { printf("Input file does not exist\n"); return 0; }
   while(TRUE) {
      nscan = fscanf(infile, "%40[^,], %ld, %f, %d%c",
                  tab[count].owner, &tab[count].balance, &tab[count].rate, &tab[count].year, &termch);
      if (nscan == EOF) break;
      if (nscan != 5 || termch != '\n') printf("Line Error in file %s \n", fileName);
      else count++;
    }
    fclose(infile);
    return count; }
```

Ανάγνωση Στοιχείων Αρχείων

- Πόσες φορές θα κληθεί η συνάρτηση;

```
newDataSize = readFileData("bank_new.dat", newData);
oldDataSize = readFileData("bank_old.dat", oldData);
```

Σύγκριση Καταθετών

- θα τυπώνει στην οθόνη όσους καταθέτες εμφανίζονται στο αρχείο bank_new.dat και **δεν** εμφανίζονται στο αρχείο bank_old.dat. Η σύγκριση να γίνει βάση του ονόματος του καταθέτη.
- Συνάρτηση compareDeposits
  - Ορίσματα?
  - Τύπος?

compareDeposits

- Συγκρίνει πίνακες newData, oldData
  - Απαραίτητα μεγέθη πινάκων
- Αποτελέσματα στον πίνακα missing
  - Πόσοι καταθέτες δεν εμφανίζονται?

```
int compareDeposits( int newS, accountT newTab[], int oldS, accountT oldTab[], accountT
missing[]){
   int i,j;
   int count = 0; int flag = 0;
   for(i=0;i<newS;i++)
   {  flag = 0;
      for(j=0;j<oldS;j++)
        {   if (strcmp(newTab[i].owner, oldTab[j].owner) == 0)
            {   flag = 1;
                break;
            }
        }
        if (flag == 0) missing[count++] = newTab[i];
   }
    return count;
}
```

Κλήση και εμφάνιση Αποτελεσμάτων

```
mSize = compareDeposits(newDataSize, newData,
                                        oldDataSize, oldData, missing);
printf(" Clients in List New not in Old \n");
for(i=0;i<mSize;i++)
  printf("- %s %ld \n", missing[i].owner, missing[i].balance);
```

maziTaFagame

- θα αποθηκεύει σε ένα αρχείο με το όνομα “mztfgm.dat” όλους τους καταθέτες που άνοιξαν λογαριασμό από το 2000 και μετά από το αρχείο bank_new.dat, με την ίδια γραμμογράφηση που είναι τα παραπάνω αρχεία.
- Συνάρτηση maziTaFagame
  - Ορίσματα?
  - Τύπος?

Συνάρτηση maziTaFagame

```
void maziTaFagame(int size, accountT newTab[ ]){
   int i;
   FILE *outfile;
   outfile = fopen("mztfgm.dat", "w");
   for(i=0;i<size;i++)
     {if (newTab[i].year >= 2000)
       fprintf(outfile,"%s, %ld, %f, %d\n",
                newTab[i].owner, newTab[i].balance, newTab[i].rate, newTab[i].year);
      }
   fclose(outfile);
}
```

Υπολογισμός του Φόρου

- θα τυπώνει στην οθόνη το 40% του αθροίσματος των καταθέσεων των στοιχείων του αρχείου bank_new.dat,
- θα τυπώνει στην οθόνη το 40% του αθροίσματος των καταθέσεων των στοιχείων του αρχείου bank_old.dat,
- Συνάρτηση taxExpected
  - Ορίσματα?
  - Τύπος?

taxExpected

```
float taxExpected(int size, accountT tab[ ]){
   int i;
   float tax = 0;
   for(i=0;i<size;i++)
     tax = tax + tab[i].balance;
   return tax * 0.4 ;
}
```

Υπολογισμός και Εμφάνιση

- Μια κλήση για κάθε σύνολο καταθετών

```
printf("Expected Tax New %.2f \n",
                        taxExpected(newDataSize, newData));

printf("Expected Tax Old %.2f \n",
                        taxExpected(oldDataSize, oldData));
```

Μεγαλύτερες Καταθέσεις

- θα τυπώνει στην οθόνη το **όνομα** του καταθέτη με την μεγαλύτερη σε ύψος κατάθεση από το αρχείο bank_new.dat καθώς **και** το **έτος** που άνοιξε λογαριασμό.
- θα τυπώνει στην οθόνη το **όνομα** του καταθέτη με την μεγαλύτερη σε ύψος κατάθεση από το αρχείο bank_old.dat καθώς **και** το **έτος** που άνοιξε λογαριασμό.
- Συνάρτηση maxBalanceOwner
  - Ορίσματα?
  - Τύπος?

Καταθέτες

- Εφόσον όλες οι εκτυπώσεις από την main:
  - Όνομα καταθέτη και έτος από bank_new.dat
    - char maxNew[40];
    - int yearNew;
  - Όνομα καταθέτη και έτος από bank_old.dat
    - char maxOld[40];
    - int yearOld;

```
void maxBalanceOwner(int size, accountT tab[], char name[], int *year){
  int i;
  long maxBalance;
  maxBalance = tab[0].balance;
  strcpy(name, tab[0].owner);
  *year =  tab[0].year;
  for(i=1;i<size;i++)
    if (tab[i].balance > maxBalance)
    {
        strcpy(name, tab[i].owner);
        *year =  tab[i].year;
        maxBalance = tab[i].balance;
    }
}
```

Κλήση & Εμφανιση

```
maxBalanceOwner(newDataSize, newData, maxNew, &yearNew );
maxBalanceOwner(oldDataSize, oldData, maxOld, &yearOld );

printf("Max in New: %s %d \n",maxNew, yearNew);
printf("Max in Old: %s %d \n",maxOld, yearOld);
```