Πίνακες

Ιωάννης Γ. Τσούλος

2020

1 Μονοδιάστατοι πίνακες

Οι πίνακες είναι δομές δεδομένων που διαθέτουν ένα πλήθος από στοιχεία του ίδιου τύπου. Για παράδειγμα η βαθμολογία σε ένα μάθημα αποθηκεύτεται σε πίνακα. Κάθε στοιχείο του πίνακα αντιπροσωπεύει την βαθμολογία ενός σπουδαστή στο μάθημα αυτό. Άλλο χαρακτηριστικό παράδειγμα χρήσης πίνακα είναι η μισθοδοσία μιας ομάδας υπαλλήλων. Οι πίνακες διαθέτουν συγκεκριμένο αριθμό στοιχείων με το πρώτο στοιχείο να βρίσκεται στην θέση 0. Η δήλωση μονοδιάστατου πίνακα γίνεται με το ακόλουθο σχήμα

```
type name[size];
```

όπου type είναι κάποιος τύπος δεδομένων που υποστηρίζει η γλώσσα (πχ. double), name είναι το όνομα του πίνακα και size είναι το πλήθος των στοιχείων του πίνακα. Αυτό το πλήθος στοιχείων φυσικά δεν μπορεί να είναι αρνητικό.

Στον αλγόριθμο 1 έχουμε χρήση πίνακα για την ανάθεση και εμφάνιση της βαθμολογίας 5 σπουδαστών. Όπως βλέπουμε κάθε στοιχείο του πίνακα θεωρείται μια ανεξάρτητη μεταβλητή από το πρόγραμμα. Η προσπέλαση στα στοιχεία του πίνακα γίνεται με την χρήση των αγκυλών. Φυσικά είναι εφικτό να αλλάξουν απευθείας τα στοιχεία ενός πίνακα χωρίς ανάγνωση όπως στον αλγόριθμο 2, όπου αλλάζει το δεύτερο και το τρίτο στοιχείο του πίνακα grade. Ωστόσο η ανάγνωση πίνακα και η ανάθεση σε στοιχεία δεν είναι ο μόνος τρόπος να μπουν τιμές σε έναν πίνακα. Στην γλώσσα C μπορούμε να αναθέσουμε τιμές σε στοιχεία ενός πίνακα και κατά την δήλωση του, όπως φαίνεται στον αλγόριθμο 3.

Στα παραδείγματα αυτά μέχρι στιγμής γίνεται εισαγωγή της βαθμολογίας απευθείας στον πίνακα χωρίς να γίνεται κάποιος έλεγχος για την εγκυρότητα των τιμών (οι βαθμοί πρέπει να είναι μεταξύ 0 και 10). Αυτό γίνεται στον αλγόριθμο 4, όπου με την βοήθεια της βοηθητικής συνάρτησης readGrade() γεμίζει ο πίνακας με έγκυρες τιμές.

Algorithm 1 Είσοδος και εμφάνισης της βαθμολογίας 5 σπουδαστών.

```
1  # include <stdio.h>
2  int main()
3  {
4          double grade[5];
5          int i;
6          for(i=0;i<5;i++)
7          {
8                  printf("Doste ton bathmo %d \n",i);
9                  scanf("%lf",&grade[i]);
10         }
11         for(i=0;i<5;i++)
12         {
13                 printf("o vathmos stin thesi  %d  einai %lf\n",
14                         i,grade[i]);
15         }
16         return 0;
17 }
```

Algorithm 2 Απευθείας ανάθεση σε στοιχεία ενός πίνακα βαθμολογίας.

```
1  # include <stdio.h>
2  int main()
3  {
4          double grade[5];
5          int i;
6          for(i=0;i<5;i++)
7          {
8                  printf("Doste ton bathmo %d\n",i);
9                  scanf("%lf",&grade[i]);
10         }
11         grade[1]=5;
12         grade[2]=3;
13         for(i=0;i<5;i++)
14         {
15           printf("o vathmos stin thesi %d einai %lf\n",i,grade[i]);
16         }
17         return 0;
18 }
```

Algorithm 3 Ανάθεση τιμών σε πίνακα κατά την δήλωση.

```
1  # include <stdio.h>
2  int main()
3  {
4          double grade[5]={10,3,4,7,8};
5          int i;
6          for(i=0;i<5;i++)
7          {
8              printf("o vathmos stin thesi %d einai %lf\n",i,grade[i]);
9          }
10         return 0;
11 }
```

Algorithm 4 Ανάγνωση βαθμολογίας σπουδαστών με έλεγχο ορίων.

```
1  # include <stdio.h>
2  double readGrade()
3  {
4          double x;
5          do
6          {
7                  printf("Doste vathmo ");
8                  scanf("%lf",&x);
9          }while(x<0 || x>10);
10         return x;
11 }
12
13 int main()
14 {
15         double grade[5];
16         int i;
17         for(i=0;i<5;i++)
18         {
19                 printf("Doste ton bathmo %d\n",i);
20                 grade[i]=readGrade();
21         }
22         for(i=0;i<5;i++)
23         {
24             printf("o vathmos stin thesi %d einai %lf\n",i,grade[i]);
25         }
26         return 0;
27 }
```

2 Αλγόριθμοι

2.1 Πίνακες σε συνάρτηση

Το επόμενο θέμα που θα εξετάσουμε είναι η χρήση πινάκων σαν ορίσματα σε συναρτήσεις. Ο απλούστερος τρόπος για να πραγματοποιηθεί αυτό είναι με την χρήση δύο ξεχωριστών ορισμάτων σε πίνακα: ένα όρισμα για τον πίνακα και ένα όρισμα για το πλήθος των στοιχείων του πίνακα. Για παράδειγμα η δήλωση:

```
void readArray(int x[],int size)
```

χρησιμοποιείται για να δηλώσουμε μια συνάρτηση που λαμβάνει έναν πίνακα ακεραίων σαν πρώτο όρισμα και έναν ακέραιο σαν δεύτερο όρισμα. Συνήθως το δεύτερο όρισμα χρησιμοποιείται για να αναπαραστήσει την διάσταση του πίνακα. Επιπλέον μια συνάρτηση που δέχεται σαν όρισμα έναν πίνακα μπορεί να αλλάξει τα στοιχεία του πίνακα και αυτές οι αλλαγές να γίνουν ορατές και στην καλούσα συνάρτηση. Στο παράδειγμα του αλγορίθμου 5 η συνάρτηση readArray διαβάζει από το πληκτρολόγιο τα στοιχεία ενός πίνακα και η συνάρτηση printArray εμφανίζει αυτά τα στοιχεία στην οθόνη. Στον αλγόριθμο αυτό χρησιμοποιούμε σαν παράδειγμα ένα πίνακα 5 στοιχείων και χρησιμοποιούμε το 5 σαν δεύτερο όρισμα στις συναρτήσεις readArray και printArray. Ωστόσο αν χρειάζεται το πρόγραμμα να γίνει πιο γενικό και να μην χρειάζονται μεγάλες αλλαγές για διαφορετικούς πίνακες μπορεί να χρησιμοποιηθεί η εντολή # define. Η χρήση της εντολής #define παρουσιάζεται στον αλγόριθμο 6. Η εντολή #define ανήκει στον προεπεξεργαστή της γλώσσας και ο σκοπός της είναι να αντικαταστήσει κάθε εμφάνιση της λέξης που είναι μετά το #define με μία άλλη έκφραση. Στην συγκεκριμένη περίπτωση αντικαθιστά το ARRAYSIZE με τον αριθμό 5.

Στην συνέχεια με χρήση συναρτήσεων παρουσιάζονται μια σειρά από βασικούς αλγόριθμους στον χώρο των πινάκων.

2.2 Αναζήτηση

Ένας από τους πιο σημαντικούς αλγόριθμους για πίνακες είναι η αναζήτηση στοιχείων. Για παράδειγμα έστω ο πίνακας

$$x = [10, 20, 7, 9, 16, 3]$$

και ζητείται η εύρεση της θέσης του αριθμού 7. Για να γίνει η αναζήτηση συγκρίνουμε κάθε στοιχείο με το 7 μέχρι να βρεθεί. Στον συγκεκριμένο πίνακα το 7 είναι στην θέση 2. Μετά την εύρεσή του ο αλγόριθμος 7 διακόπτεται.

Μια άλλη ενδιαφέρουσα συνάρτηση είναι η καταμέτρηση στοιχείων που ικανοποιούν κάποιο κριτήριο σε έναν πίνακα. Για παράδειγμα να μπορέσουν να βρεθούν τα θετικά στοιχεία σε έναν πίνακα, όπως παρουσιάζεται στον αλγόριθμο 8.

2.3 Μέγιστο - ελάχιστο

Η εύρεση μεγίστου (ή και ελαχίστου) σε έναν πίνακα είναι επίσης μια σημαντική διαδικασία στους πίνακες. Για να πραγματοποιηθεί η αναζήτηση του μεγαλύτερου σε

Algorithm 5 Εισαγωγή και εμφάνιση πίνακα.

```
1  # include <iostream.h>
2
3  void     readArray(int x[],int size)
4  {
5          int i;
6          for(i=0;i<size;i++)
7          {
8                  printf("Enter element for %d\n",i);
9                  scanf("%d",&x[i]);
10         }
11 }
12
13 void     printArray(int x[],int size)
14 {
15         int i;
16         for(i=0;i<size;i++)
17         {
18                 printf("element at %d is %d\n",i,x[i]);
19         }
20 }
21
22 int main()
23 {
24         int x[5];
25         readArray(x,5);
26         printArray(x,5);
27         return 0;
28 }
```

Algorithm 6 Είσοδος/έξοδος πίνακα και χρήση της εντολής #define.

```
1  # include <stdio.h>
2  # define ARRAYSIZE       5
3
4  void     readArray(int x[],int size)
5  {
6          int i;
7          for(i=0;i<size;i++)
8          {
9                  printf("Enter element for %d\n",i);
10                 scanf("%d",&x[i]);
11         }
12 }
13
14 void     printArray(int x[],int size)
15 {
16         int i;
17         for(i=0;i<size;i++)
18         {
19                 printf("element at %d is %d\n",i,x[i]);
20         }
21 }
22
23 int main()
24 {
25         int x[ARRAYSIZE];
26         readArray(x,ARRAYSIZE);
27         printArray(x,ARRAYSIZE);
28         return 0;
29 }
```

Algorithm 7 Συνάρτηση και πρόγραμμα για την αναζήτηση της πρώτης εμφάνισης στοιχείου σε πίνακα.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Auti i synartisi epistrefei tin proti thesi
5      tis timis value ston pinaka x. An den yparxei
6      epistrefei -1
7  **/
8  int search(int x[],int size,int value)
9  {
10         int i;
11         for(i=0;i<size;i++)
12         {
13                 if(x[i]==value)
14                         return i;
15         }
16         /** An teleiosei i epanalipsi, tote den
17             yparxei to stoixeio ston pinaka **/
18         return -1;
19 }
20
21 void    readArray(int x[],int size)
22 {
23         int i;
24         for(i=0;i<size;i++)
25         {
26                 printf("Enter element for %d\n",i);
27                 scanf("%d",&x[i]);
28         }
29 }
30
31 int main()
32 {
33         int array[ARRAYSIZE];
34         int value;
35         int pos;
36         readArray(array,ARRAYSIZE);
37         printf("Doste timi ");
38         scanf("%d",&value);
39         pos=search(array,ARRAYSIZE,value);
40         printf("Brethike stin thesi %d\n",pos);
41         return 0;
42 }
```

Algorithm 8 Καταμέτρηση θετικών στοιχείων σε πίνακα.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Synartisi anagnosis apo pinaka **/
5  void     readArray(int x[],int size)
6  {
7           int i;
8           for(i=0;i<size;i++)
9           {
10                  printf("Enter element for %d\n",i);
11                  scanf("%d",&x[i]);
12          }
13 }
14
15 /** Synartisi eyresis thetikon stoixeion **/
16 int countPositive(int x[],int size)
17 {
18          int count=0;
19          int i;
20          for(i=0;i<size;i++)
21                  if(x[i]>0) count++;
22          return count;
23 }
24
25 int main()
26 {
27          int array[ARRAYSIZE];
28          int total;
29          readArray(array,ARRAYSIZE);
30          total=countPositive(array,ARRAYSIZE);
31          printf("Synolikoi thetikoi arithmoi %d\n",total);
32          return 0;
33 }
```

έναν πίνακα ξεκινούμε θεωρώντας πως το μεγαλύτερο είναι το πρώτο στοιχείο του πίνακα και στην συνέχεια εξετάζουμε αν αυτός ο ισχυρισμός ευσταθεί συγκρίνοντας το τρέχον μεγαλύτερο με κάθε στοιχείο του πίνακα. Αν κάπου βρεθεί μεγαλύτερο στοιχείο αναπροσαρμόζεται αυτός ο ισχυρισμός. Πρακτικά αυτή η διαδικασία παρουσιάζεται στον αλγόριθμο 9. Αν θέλουμε η συνάρτηση maxElement να επιστρέψει το ελάχιστο στοιχείο θα χρησιμοποιηθεί ο τελεστής $<$ αντί για $>$. Ένας άλλος αλγόριθμος που συσχετίζεται με την εύρεση μεγίστου είναι αυτός της εύρεσης της θέσης του μεγίστου σε έναν πίνακα. Για να γίνει αυτό παράλληλα με το μέγιστο σε κάθε επανάληψη αναπροσαρμόζεται και η θέση του, όπως παρουσιάζεται στον αλγόριθμο 10.

2.4 Μέσος όρος

Η εύρεση μέσου όρου σε ένα πίνακα χρησιμοποιείται σε πολλές περιπτώσεις όπως σε βαθμολογία ενός μαθήματος. Για την εύρεση του μέσου όρου αθροίζονται όλα τα στοιχεία ενός πίνακα και γίνεται διαίρεση με το πλήθος τους. Πρέπει να επισημανθεί πως ο μέσος όρος είναι πάντα δεκαδικός αριθμός. Στον αλγόριθμο 11 παρουσιάζεται η εύρεση του μέσου όρου της βαθμολογίας 5 μαθητών σε ένα μάθημα.

2.5 Ταξινόμηση

Με την ταξινόμηση επιτυγχάνεται η μετατροπή ενός πίνακα από αριθμούς χωρίς κάποια δομή σε σειρά αριθμών που έχουν κάποια διάταξη, όπως για παράδειγμα να βρίσκονται σε αύξουσα σειρά. Ο απλούστερος τρόπος για να γίνει αυτό είναι με τον αλγόριθμο bubble sort που παρουσιάζεται στον αλγόριθμο 12.

3 Αλφαριθμητικά

Τα αλφαριθμητικά είναι ιδιαίτερη περίπτωση μονοδιάστατου πίνακα καθώς αποτελούνται από γράμματα και ο σκοπός τους είναι να αναπαραστήσουν λέξεις και φράσεις. Για παράδειγμα στην εντολή

```
cout<<"Helo world";
```

η έκφραση Hello world είναι ένα αλφαριθμητικό. Τα αλφαριθμητικά βρίσκονται πάντα μέσα σε διπλά εισαγωγικά και αυτό τα διαχωρίζει από τους απλούς χαρακτήρες που είναι ένα γράμμα μόνον και βρίσκονται μέσα σε μονά εισαγωγικά.

3.1 Μεταβλητές χαρακτήρα

Οι μεταβλητές χαρακτήρα χρησιμοποιούνται για να δηλώσουν ένα γράμμα όπως είναι πχ το A, το # κτλ. Οι μεταβλητές αυτές δηλώνονται σαν char αλλά για την γλώσσα C μπορούν να χρησιμοποιηθούν και σαν ακέραιοι όπως φαίνεται στο παράδειγμα 13. Από το παράδειγμα αυτό είναι φανερό πως οι μεταβλητές τύπου

Algorithm 9 Εύρεση μεγίστου σε πίνακα.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Synartisi anagnosis apo pinaka **/
5  void     readArray(int x[],int size)
6  {
7           int i;
8           for(i=0;i<size;i++)
9           {
10                  printf("Enter element for %d\n",i);
11                  scanf("%d",&x[i]);
12          }
13 }
14
15 /** Synartisi eyresis megistou **/
16 int      maxElement(int x[],int size)
17 {
18          int max=x[0];
19          int i;
20          for(i=1;i<size;i++)
21          {
22                  if(x[i]>max) max=x[i];
23          }
24          return max;
25 }
26
27 int main()
28 {
29          int array[ARRAYSIZE];
30          int maxvalue;
31          readArray(array,ARRAYSIZE);
32          maxvalue=maxElement(array,ARRAYSIZE);
33          printf("Megisto stoixeio ston pinaka einai %d\n",maxvalue);
34          return 0;
35 }
```

Algorithm 10 Εύρεση θέσης μεγίστου σε πίνακα.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Synartisi anagnosis apo pinaka **/
5  void     readArray(int x[],int size)
6  {
7          int i;
8          for(i=0;i<size;i++)
9          {
10                 printf("Enter element for %d\n",i);
11                 scanf("%d",&x[i]);
12         }
13 }
14
15 /** Synartisi eyresis thesis megistou **/
16 int      maxPosition(int x[],int size)
17 {
18         int max=x[0];
19         int maxpos=0;
20         int i;
21         for(i=1;i<size;i++)
22         {
23                 if(x[i]>max)
24                 {
25                         max=x[i];
26                         maxpos=i;
27                 }
28         }
29         return maxpos;
30 }
31
32 int main()
33 {
34         int array[ARRAYSIZE];
35         int pos,value;
36         readArray(array,ARRAYSIZE);
37         pos=maxPosition(array,ARRAYSIZE);
38         value=array[pos];
39         printf("To megisto einai stin thesi %d  kai exei timi %d\n",pos,value
40         return 0;
41 }
```

Algorithm 11 Εύρεση μέσου όρου σε πίνακα βαθμολογίας.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Synartisi anagnosis apo pinaka **/
5  void     readArray(double x[],int size)
6  {
7           int i;
8           for(i=0;i<size;i++)
9           {
10                  printf("Enter element for %d\n",i);
11                  scanf("%lf",&x[i]);
12          }
13 }
14
15 double   avgArray(double x[],int size)
16 {
17          double sum=0.0;
18          int i;
19          for(i=0;i<size;i++) sum=sum+x[i];
20          return sum/size;
21 }
22
23 int main()
24 {
25          double   grade[ARRAYSIZE];
26          double average;
27          readArray(grade,ARRAYSIZE);
28          average=avgArray(grade,ARRAYSIZE);
29          printf("O mesos oros einai %lf\n",average);
30          return 0;
31 }
```

Algorithm 12 Ταξινόμηση πίνακα.

```
1  # include <stdio.h>
2  # define ARRAYSIZE 5
3
4  /** Synartisi anagnosis apo pinaka **/
5  void     readArray(int x[],int size)
6  {
7           int i;
8           for(i=0;i<size;i++)
9           {
10                  printf("Enter element for %d\n",i);
11                  scanf("%d",&x[i]);
12          }
13 }
14
15 /** Synartisi emfanisis pinaka **/
16 void printArray(int x[],int size)
17 {
18          int i;
19          for(i=0;i<size;i++)
20                  printf("%d\n",x[i]);
21 }
22
23 /** Sinartisi taxinomisis pinaka **/
24 void     sortArray(int x[],int size)
25 {
26          int i,j,t;
27          for(i=0;i<size;i++)
28          {
29           for(j=0;j<size-1;j++)
30           {
31              if(x[j+1]<x[j])
32              {
33                t=x[j];
34                x[j]=x[j+1];
35                x[j+1]=t;
36              }
37           }
38          }
39 }
40
41 int main()
42 {
43          int array[ARRAYSIZE];
44          readArray(array,ARRAYSIZE);
45          sortArray(array,ARRAYSIZE);
46          printArray(array,ARRAYSIZE);
47          return 0;
48 }
```

Algorithm 13 Παράδειγμα εμφάνισης γραμμάτων και ASCII κωδικών.

```
1  # include <stdio.h>
2  int main()
3  {
4          char letter;
5          int code;
6          char nextLetter;
7          int nextCode;
8          letter='A';
9          code=letter;
10         nextLetter=letter+1;
11         nextCode=nextLetter;
12         printf("To proto gramma %c\n",letter);
13         printf("O protos kodikos %d\n",code);
14         printf("To epomeno gramma %c\n",nextLetter);
15         printf("O epomenos kodikos %d\n",nextCode);
16         return 0;
17 }
```

char μπορούν να χρησιμοποιηθούν και σαν ακέραιοι αριθμοί με προσθέσεις, αναθέσεις κτλ. Επιπλέον οι κωδικοί 65 και 66 που εμφανίζοναι για τα γράμματα A και B είναι οι αντίστοιχοι κωδικοί αυτών των γραμμάτων στον πίνακα ASCII.

3.2 Γράμματα από μικρά σε κεφαλαία

Στον πίνακα ASCII τα κεφαλαία γράμματα βρίσκονται πριν από τα μικρά στις θέσεις 65..89. Τα μικρά γράμματα βρίσκονται μετά την θέση 97. Επιπλέον τα γράμματα είναι συνεχόμενα, που σημαίνει πως για παράδειγμα το B βρίσκεται μετά το γράμμα A. Στον αλγόριθμο 14 εκμεταλλευόμαστε αυτήν την κατάσταση και γίνεται μετατροπή κάθε μικρού γράμματος στο αντίστοιχο κεφαλαίο. Η ανάγνωση των γραμμάτων γίνεται επαναληπτικά και σταματά με την είσοδο του ειδικού γράμματος #.

3.3 Πίνακες χαρακτήρων

Βάζοντας πολλά γράμματα μαζί σε έναν πίνακα δημιουργείται ένα αλφαριθμητικό για την αναπαράσταση λέξεων και προτάσεων. Ωστόσο ο προγραμματιστής πρέπει να ξέρει που βρίσκεται το τέλος ενός αλφαριθμητικού μέσα σε έναν πίνακα. Αυτό γίνεται με την χρήση του ειδικού χαρακτήρα '\0' για τον καθορισμό του τέλους. Προσοχή πρέπει να δοθεί στο γεγονός πως αυτός ο χαρακτήρας (NULL) δεν είναι το '0' αλλά το γράμμα που βρίσκεται στην θέση 0 του πίνακα ASCII, το οποίο είναι μη εκτυπώσιμο. Το πρόγραμμα στον αλγόριθμο 15 γεμίζει τον πίνακα γραμμάτων X και τον εμφανίζει στην οθόνη. Φυσικά αυτός ο τρόπος δεν είναι ιδιαίτερα βολικός, αφού πρέπει ο χρήστης να εισάγει ένα προς ένα τα γράμματα του αλφαριθμητικού αλλά και τον χαρακτήρα τερματισμό. Το πιο απλό που μπορεί

Algorithm 14 Πρόγραμμα μετατατροπής των μικρών γραμμάτων σε κεφαλαία.

```
1  # include <stdio.h>
2  int main()
3  {
4          char letter;
5          char uppercase;
6          do
7          {
8                  printf("Doste gramma # gia termatismo \n");
9                  scanf("%c",&letter);
10                 if(letter>='a' && letter<='z')
11                 {
12                         uppercase=letter-('a'-'A');//32
13                         printf("Kefalaio gramma %c\n",uppercase);
14                 }
15                 else
16                         printf("Eidiko gramma %c\n",letter);
17         }while(letter!='#');
18         return 0;
19 }
```

να κάνει κανείς εναλλακτικά είναι να εισάγει το αλφαριθμητικό με την χρήση του cin, όπως παρουσιάζεται στον αλγόριθμο 16. Ωστόσο η cin έχει το πρόβλημα πως μπορεί να διαβάσει μόνον αλφαριθμητικά μιας λέξής και όχι προτάσεις. Σε αυτήν την περίπτωση μπορεί να χρησιμοποιηθεί η συνάρτηση gets() από την βιβλιοθήκη stdio.h που παρουσιάζεται στον αλγόριθμο 17.

3.4 Συναρτήσεις αλφαριθμητικών

Στην συνέχεια παρουσιάζονται δύο χρήσιμες συναρτήσεις αλφαριθμητικών για αντιγραφή αλφαριθμητικών και εύρεση μήκους αντίστοιχα. Για την χρήση αυτών των συναρτήσεων απαιτείται η συμπερίληψη της βιβλιοθήκης string.h

3.4.1 Η συνάρτηση strcpy

Η συνάρτηση strcpy χρησιμοποιείται για την αντιγραφή ενός αλφαριθμητικού σε ένα άλλο. Δεν μπορεί να χρησιμοποιηθεί απευθείας ανάθεση, καθώς τα αλφαριθμητικά είναι πίνακες και σαν πίνακες δεν μπορούμε να τους αναθέσουμε ποτέ απευθείας μια τιμή. Στο παράδειγμα 18 παρουσιάζεται η χρήση της συνάρτησης strcpy και παρουσιάζεται επίσης και μια διαφορετική υλοποίησή της με το όνομα strcopy. Η συνάρτηση strcopy επιστρέφει και το πλήθος των γραμμάτων που αντιγράφηκαν.

Algorithm 15 Εισαγωγή γραμμάτων σε αλφαριθμητικό με διαδοχικές αναθέσεις.

```
1  # include <stdio.h>
2
3  int main()
4  {
5          char X[100];
6          X[0]='A';
7          X[1]='R';
8          X[2]='T';
9          X[3]='A';
10         X[4]='\0';
11         printf("X is %s\n",X);
12         return 0;
13 }
```

Algorithm 16 Εισαγωγή αλφαριθμητικού με χρήση του cin.

```
1  # include <stdio.h>
2
3  int main()
4  {
5          char X[100];
6          printf("Doste to X \n");
7          scanf("%s",X);
8          printf("X is %s\n",X);
9          return 0;
10 }
```

Algorithm 17 Ανάγνωση αλφαριθμητικού με χρήση της gets.

```
1  # include <stdio.h>
2
3  int main()
4  {
5          char X[100];
6          printf("Doste to X \n");
7          gets(X);
8          printf("X is %s\n",X);
9          return 0;
10 }
```

Algorithm 18 Παρουσίαση της συνάρτησης strcpy.

```
1  # include <string.h>
2  # include <stdio.h>
3
4  /** Synartisi antigrafis tou a sto b **/
5  int strcopy(char b[],char a[])
6  {
7          int i;
8          int j=0;
9          for(i=0;a[i]!='\0';i++)
10         {
11                 b[j]=a[i];
12                 j=j+1;
13         }
14         b[j]='\0';
15         //epistrofi tou plithous ton grammaton
16         return j;
17 }
18
19 int main()
20 {
21         char x1[100],x2[100];
22         int length;
23         printf("Doste mia lexi \n");
24         scanf("%s",x1);
25         strcpy(x2,x1);
26         printf("To x1 einai %s\n",x1);
27         printf("To x2 einai %s\n",x2);
28         strcpy(x1,"This is a test ");
29         length=strcopy(x2,x1);
30         printf("To x1 einai %s\n",x1);
31         printf("To x2 einai %s\n",x2);
32         printf("Plithos grammaton %d\n",length);
33         return 0;
34 }
```

Algorithm 19 Η συνάρτηση strlen.

```
1  # include <string.h>
2  # include <stdio.h>
3
4  int mystrlen(char x[])
5  {
6          int count=0;
7          while(x[count]!='\0') count++;
8          return count;
9  }
10
11 int main()
12 {
13         char x[100];
14         int len;
15         printf("Doste mia frasi \n");
16         gets(x);
17         len=strlen(x);
18         printf("To mikos einai %d\n",len);
19         len=mystrlen(x);
20         cout<<"To mikos einai "<<len<<endl;
21         return 0;
22 }
```

3.4.2 Η συνάρτηση strlen

Η επόμενη συνάρτηση που χρησιμοποιείται τακτικά στον προγραμματισμό με αλφαριθμητικά είναι η strlen, που επιστρέφει το πλήθος των γραμμάτων ενός αλφαριθμητικού. Στο παράδειγμα 19 παρουσιάζεται η ενσωματωμένη συνάρτηση strlen καθώς και μια εναλλακτική υλοποίησή της.

4 Διδιάστατοι πίνακες

Η γενική δήλωση διδιάστατων πινάκων στην C είναι

```
type name[rows][columns];
```

όπου type είναι ο τύπος του πίνακα (πχ. double), name είναι το όνομά του, rows είναι το πλήθος των γραμμών και columns είναι το πλήθος των στηλών. Εσωτερικά στην γλώσσα οι διδιάστατοι πίνακες είναι πίνακες πινάκων και όχι αναπαραστάσεις σε διδιάστατη μορφή.

Algorithm 20 Ανάγνωση και εμφάνιση διδιάστατου πίνακα.

```
1  # include <stdio.h>
2  # define ROWS 3
3  # define COLS 2
4
5  int main()
6  {
7          int a[ROWS][COLS];
8          int i,j;
9          for(i=0;i<ROWS;i++)
10         {
11                 for(j=0;j<COLS;j++)
12                 {
13                         printf("Doste␣to␣stoixeio␣%d,%d\n",i,j);
14                         scanf("%d",&a[i][j]);
15                 }
16         }
17         printf("Emfanisi␣pinaka␣\n");
18         for(i=0;i<ROWS;i++)
19         {
20                 for(j=0;j<COLS;j++)
21                 {
22                         printf("%d␣",a[i][j]);
23                 }
24                 printf("\n");
25         }
26         return 0;
27 }
```

4.1 Είσοδος τιμών

Μιας και οι διδιάστατοι πίνακες διαθέτουν αριθμό γραμμών και στηλών είναι απαραίτητο να χρησιμοποιηθεί διπλήε επανάληψη για την προσπέλαση τους. Το παράδειγμα του αλγορίθμου 20 παρουσιάζει την ανάγνωση και εμφάνιση ενός πίνακα ακεραίων αριθμών. Παρόμοια μπορούμε να χρησιμοποιήσουμε διδιάστατους πίνακες για να εμφανίσουμε τον πίνακα προπαίδειας, όπως παρουσιάζεται στον αλγόριθμο 21.

4.2 Πρόσθεση πινάκων

Ένας ακόμα χρήσιμος αλγόριθμος στους πίνακες δύο διαστάσεων είναι η πρόσθεση πινάκων στοιχείο προς στοιχείο. Φυσικά όπως γνωρίζουμε για να προστεθούν δύο πίνακες δύο διαστάσεων θα πρέπει οι γραμμές και οι στήλες να είναι ίδιες. Στο παράδειμα 22 παρουσιάζεται η πρόσθεση δύο ίδιων πινάκων.

Algorithm 21 Πίνακας προπαίδειας.

```
1  # include <stdio.h>
2  # define ROWS 10
3  # define COLS 10
4
5  int main()
6  {
7          int a[ROWS][COLS];
8          int i,j;
9          for(i=0;i<ROWS;i++)
10         {
11                 for(j=0;j<COLS;j++)
12                 {
13                         a[i][j]=(i+1)*(j+1);
14                 }
15         }
16         printf("Emfanisi pinaka \n");
17         for(i=0;i<ROWS;i++)
18         {
19                 for(j=0;j<COLS;j++)
20                 {
21                         printf("%d\t",a[i][j]);
22                 }
23                 printf("\n");
24         }
25         return 0;
26 }
```

Algorithm 22 Παράδειγμα πρόσθεσης πινάκων.

```
1  # include <stdio.h>
2  # define ROWS 3
3  # define COLS 2
4
5  int main()
6  {
7          int a[ROWS][COLS];
8          int b[ROWS][COLS];
9          int c[ROWS][COLS];
10         int i,j;
11         for(i=0;i<ROWS;i++)
12         {
13                 for(j=0;j<COLS;j++)
14                 {
15                         printf("Doste to stoixeio gia ton a %d,%d\n",i,j);
16                         scanf("%d",&a[i][j]);
17                         printf("Doste to stoixeio gia ton b %d,%d\n",i,j);
18                         scanf("%d",&b[i][j]);
19                 }
20         }
21         for(i=0;i<ROWS;i++)
22                 for(j=0;j<COLS;j++)
23                         c[i][j]=a[i][j]+b[i][j];
24
25         for(i=0;i<ROWS;i++)
26         {
27                 for(j=0;j<COLS;j++)
28                 {
29                         printf("%d\t",c[i][j]);
30                 }
31                 printf("\n");
32         }
33         return 0;
34 }
```

4.3 Μεγαλύτερο στοιχείο ανά γραμμή

Ας υποθέσουμε πως αποθηκεύουμε σε έναν διδιάστατο πίνακα τις επιδόσεις 5 μαθητών σε 3 μαθήματα. Στην συνέχεια ζητείται να βρούμε τον σπουδαστή που έλαβε την μεγαλύτερη βαθμολογία ανά μάθημα. Αυτό παρουσιάζετα στον αλγόριθμο 23.

Algorithm 23 Μεγαλύτερο στοιχείο ανά γραμμή.

```
1  # include <stdio.h>
2  # define STUDENTS 5
3  # define LESSONS  3
4
5  double readGrade()
6  {
7          double x;
8          do
9          {
10                 printf("Doste vathmo ");
11                 scanf("%lf",&x);
12         }while(x<0 || x>10);
13         return x;
14 }
15
16 int main()
17 {
18         double grade[STUDENTS][LESSONS];
19         int i,j;
20         for(i=0;i<STUDENTS;i++)
21         {
22                 for(j=0;j<LESSONS;j++)
23                 {
24                         printf("Doste bathmo gia thesi %d,%d\n",i,j);
25                         grade[i][j]=readGrade();
26                 }
27         }
28         for(j=0;j<LESSONS;j++)
29         {
30                 int maxpos=0;
31                 double maxvalue=grade[maxpos][j];
32                 for(i=0;i<STUDENTS;i++)
33                 {
34                         if(grade[i][j]>maxvalue)
35                         {
36                                 maxvalue=grade[i][j];
37                                 maxpos=i;
38                         }
39                 }
40                 printf("Sto mathima  %d Ton megalytero
41 ~~~~~~~~~~~~~~~~~~~~~~~~vathmo Exei o mathitis %d \n",i,maxpos);
42         }
43         return 0;
44 }
```