PROJECT ΔΟΜΕΣ ΔΕΔΟΜΕΝΩΝ

ΑΝΔΡΕΑΣ ΚΑΤΣΑΡΟΣ ΑΜ 1084522

ΓΡΗΓΟΡΗΣ ΤΖΩΡΤΖΑΚΗΣ ΑΜ 1084538

ΑΝΤΩΝΗΣ ΠΟΜΟΝΗΣ ΑΜ1084616

ΓΙΑΝΝΗΣ ΜΑΡΓΕΤΗΣ ΑΜ 1084521

Σημαντικο:

-Στα παρακάτω screenshot και στους κώδικές μας όπως μπορείτε να δείτε , έχουμε μετονομάσει το αρχείο από ocean.csv σε project.csv επομένως σε περίπτωση που θέλετε να τρέξετε το αρχείο με βάση τα προγράμματα μας παρακαλούμε μετονομάστε το αρχείο σε project.csv.Στο zip μας με τους κώδικες και την αναφορά παραθέτουμε οι ίδιοι το csv με την ορθή ονομασία.

PART 1

```
typedef struct Okeanos
{
    int mhnas, mera, xronos;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;

}Okeanos;
```

Χρησιμοποιουμε την δομη οκεανος με παραμετρους τις int τιμες μηνας μερα χρονος και τις float τιμες thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen.

```c
int FileOpener (FILE* f)
{
int counter = 0;
char c;
for (c = fgetc(f); c != '\n' && c != EOF; c=fgetc(f)){};
for (c=fgetc(f); c!=EOF; c=fgetc(f))
{
    if (c == '\n')
    {
        counter++;
    }
}
    counter++;
    return counter;
}
```

Χρησιμοποιουμε την συναρτηση file opener με ορισμα F το αρχειο csv που θελουμε να εισαγουμε στο προγραμμα.

```c
int FileOpener(FILE* f);
void insertionSort(Okeanos c[], int n);
void printArray(Okeanos c[], int n);
int main(){
    FILE *f;
    f = fopen("project.csv","r+");
    if (f == NULL) {exit(1);}

    int count=FileOpener(f);

    f = fopen("project.csv","r+");

    Okeanos OkeanosArray[count];
    int i=0;
    int mhnas, mera, xronos;
    mhnas=mera=xronos=0;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;
    thermokrasia=phosphate=silicate=nitrite=nitrate=salinity=oxygen=0.0;
    char c;
    for (c = fgetc(f); c != '\n' && c != EOF; c=fgetc(f)){};


    while (c!=EOF)
    {
        fscanf(f, "%d/%d/%d, %f, %f, %f, %f, %f, %f, %f", &mhnas, &mera, &xronos, &thermokrasia, &phosphate, &silicate, &nitrite, &nitrate, &salinity, &oxygen);
        OkeanosArray[i].mhnas=mhnas;
        OkeanosArray[i].mera=mera;
        OkeanosArray[i].xronos=xronos;
        OkeanosArray[i].thermokrasia=thermokrasia;
        OkeanosArray[i].phosphate=phosphate;
        OkeanosArray[i].silicate=silicate;
        OkeanosArray[i].nitrite=nitrite;
        OkeanosArray[i].nitrate=nitrate;
        OkeanosArray[i].salinity=salinity;
        OkeanosArray[i].oxygen=oxygen;
        c=getc(f);
        if (c == '\n') {i++;}
    }

    /*
    for(i = 0; i<count; i++){
        printf("%d / %d / %d \n", OkeanosArray[i].mhnas , OkeanosArray[i].mera , OkeanosArray[i].xronos);

    } */

  insertionSort(OkeanosArray,count);
  printArray(OkeanosArray,count);
}
```

Αρχικα καλουμε στην main την συναρτηση file opener με σκοπό το αρχειο project.csv .Αφου γινει η αναγνωση του αρχειου δημιουργουμε τον πινακα τυπου ωκεανος με ονομα OkeanosArray καια στοιχεια count τα οποια εχουν μετρηθει με βαση το αρχειο.

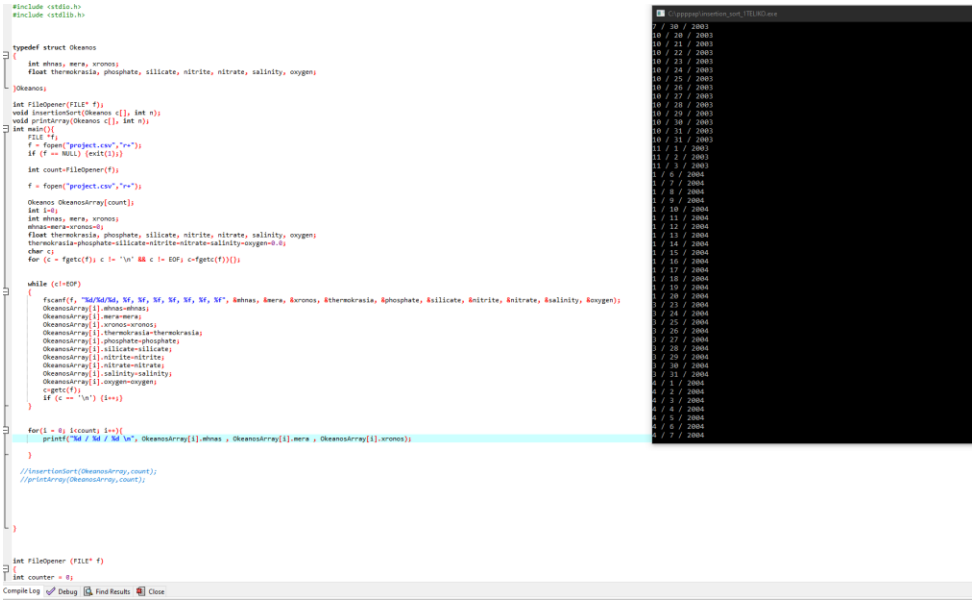
Οριζουμε τις μεταβλητες int mhnas, mera, xronos και float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen; στις οποιες περναμε αρχικα τις τιμες απο το αρχειο.

Μεσω της εντολης fscanf και του while loop περναμε καταλληλες τιμες απο το αρχειο στις αντιστοιχες θεσεις τους στον πινακα ωκεανου. Η καθε θεση του πινακα i περιεχει καποιες τιμες για μια συγκεκριμενη ημερομηνια.

Πλεον εχουμε περασει τις ημερομηνιες και τις τιμες μας στον πινακα OkeanosArray.

Παραθετουμε ενα κομματι της αρχικης εκτελεσης του προγραμματος.

Μεσω της printf εκτυπωνουμε τον μηνα, την μερα και τον χρονο ενος μερους του αρχειου.



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


typedef struct Okeanos
{
    int mhnas, mera, xronos;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;

}Okeanos;

int FileOpener(FILE* f);
void insertionSort(Okeanos c[], int n);
void printArray(Okeanos c[], int n);
int main(){
    FILE *f;
    f = fopen("project.csv","r+");
    if (f == NULL) {exit(1);}

    int count=FileOpener(f);

    f = fopen("project.csv","r+");

    Okeanos OkeanosArray[count];
    int i=0;
    int mhnas, mera, xronos;
    mhnas=mera=xronos=0;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;
    thermokrasia=phosphate=silicate=nitrite=nitrate=salinity=oxygen=0.0;
    char c;
    for (c = fgetc(f); c != '\n' && c != EOF; c=fgetc(f)){}


    while (c!=EOF)
    {
        fscanf(f, "%d/%d/%d, %f, %f, %f, %f, %f, %f, %f", &mhnas, &mera, &xronos, &thermokrasia, &phosphate, &silicate, &nitrite, &nitrate, &salinity, &oxygen);
        OkeanosArray[i].mhnas=mhnas;
        OkeanosArray[i].mera=mera;
        OkeanosArray[i].xronos=xronos;
        OkeanosArray[i].thermokrasia=thermokrasia;
        OkeanosArray[i].phosphate=phosphate;
        OkeanosArray[i].silicate=silicate;
        OkeanosArray[i].nitrite=nitrite;
        OkeanosArray[i].nitrate=nitrate;
        OkeanosArray[i].salinity=salinity;
        OkeanosArray[i].oxygen=oxygen;
        c=getc(f);
        if (c == '\n') {i++;}
    }

    for(i = 0; i<count; i++){
        printf("%d / %d / %d \n", OkeanosArray[i].mhnas , OkeanosArray[i].mera , OkeanosArray[i].xronos);
    }

   //insertionSort(OkeanosArray,count);
   //printArray(OkeanosArray,count);



}


int FileOpener (FILE* f)
{
int counter = 0;
```

```
7 / 30 / 2003
10 / 20 / 2003
10 / 21 / 2003
10 / 22 / 2003
10 / 23 / 2003
10 / 24 / 2003
10 / 25 / 2003
10 / 26 / 2003
10 / 27 / 2003
10 / 28 / 2003
10 / 29 / 2003
10 / 30 / 2003
10 / 31 / 2003
10 / 31 / 2003
11 / 1 / 2003
11 / 2 / 2003
11 / 3 / 2003
1 / 6 / 2004
1 / 7 / 2004
1 / 8 / 2004
1 / 9 / 2004
1 / 10 / 2004
1 / 11 / 2004
1 / 12 / 2004
1 / 13 / 2004
1 / 14 / 2004
1 / 15 / 2004
1 / 16 / 2004
1 / 17 / 2004
1 / 18 / 2004
1 / 19 / 2004
1 / 20 / 2004
3 / 23 / 2004
3 / 24 / 2004
3 / 25 / 2004
3 / 26 / 2004
3 / 27 / 2004
3 / 28 / 2004
3 / 29 / 2004
3 / 30 / 2004
3 / 31 / 2004
4 / 1 / 2004
4 / 2 / 2004
4 / 3 / 2004
4 / 4 / 2004
4 / 5 / 2004
4 / 6 / 2004
4 / 7 / 2004
```

Insertion Sort:

Στην συνεχεια υλοποιουμε την συναρτηση insertionsort με ορισματα τον πινακα μας και το μεγεθος του.

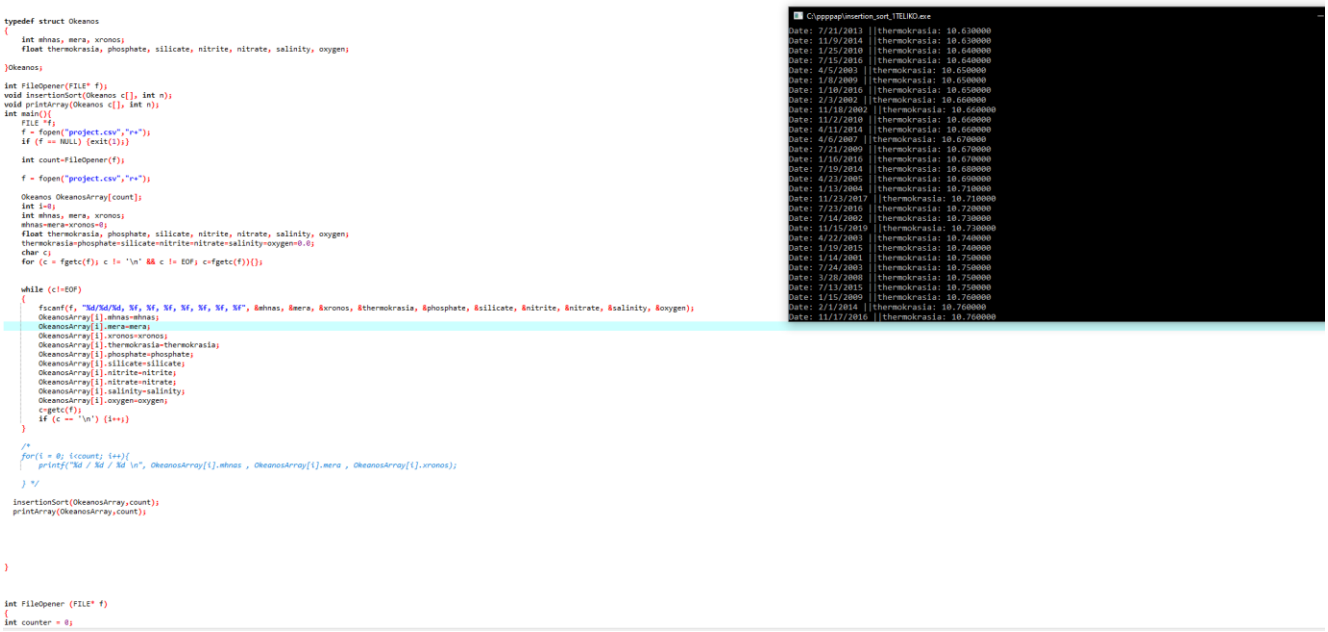
```
//INSERTION SORT
void insertionSort(Okeanos c[], int n)
{
    int i,  j;
    float key,key1,key2,key3;
    for (i = 1; i < n; i++) {
        key = c[i].thermokrasia;
        key1=c[i].mera;
        key2=c[i].mhnas;
        key3=c[i].xronos;
        j = i - 1;


        while (j >= 0 && c[j].thermokrasia > key) {
            c[j+1].thermokrasia = c[j].thermokrasia;
            c[j+1].mera = c[j].mera;
            c[j+1].mhnas = c[j].mhnas;
            c[j+1].xronos = c[j].xronos;
            j = j - 1;
        }
       c[j+1].thermokrasia = key;
        c[j+1].mera = key1;
        c[j+1].mhnas = key2;
        c[j+1].xronos = key3;
    }
}
```

Η insertion sort εκτελεί συνεχόμενες επαναλήψεις αλλάζοντας ένα στοιχείο εισόδου σε κάθε επανάληψη και δημιουργώντας με αυτόν τον τρόπο μία ταξινομημένη λίστα εξόδου. Σε κάθε επανάληψη, η ταξινόμηση με εισαγωγή μετακινεί ένα στοιχείο από τη λίστα εισόδου, αφού έχει βρει την τοποθεσία που ανήκει η ταξινομημένη λίστα και το εισάγει εκεί. Τη διαδικασία αυτή την επαναλαμβάνει μέχρι να μην υπάρχουν άλλα στοιχεία εισόδου.

Χρησιμοποιουμε τον πινακα Okeanos c[] και εφαρμοζουμε τον παραπανω αλγοριθμο με βαση την θερμοκρασια. Χρησιμοποιουμε επισης την for loop και τις μεταβλητες keys για να αποθηκευσουμε μια συγκεκριμενη ημερομηνια(τον μηνα, την μερα , τον χρονο και την θερμοκρασια στο αντίστοιχο key).Μέσω την while loop j συγκρίνουμε το περιεχόμενο της διεύθυνσης μιας συγκεκιμενης ημερομηνίας στον πίνακ ωκεανού με το key μας.Αν βρεθει θερμοκρασια η οποία είναι μεγαλύτερη απο το key(το οποίο περιέχει αποθηκευμένο το περιεχόμενο της διεύθυνσσης της σύγκρισής μας) τότε ανταλλάσει τα περιεχόμενα του μήνα, της μέρας και του χρόνου.

Καλούμε στην main την παραπάνω συνάρτηση με ορίσματα τα OkeanosArray και count .Η insertionsort καλείται και θέτει σε αύξουσα σειρα τα στοιχεία μ εβάση την θερμοκρασία.

Καλούμε την PrintArray (με τα αντίστοιχα ορίσματα) μεσω της οποιας φαίνεται ενα κομμάτι της νέας εκτύπωσης του πίνακα.



```
typedef struct Okeanos
{
    int mhnas, mera, xronos;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;

}Okeanos;

int FileOpener(FILE* f);
void insertionSort(Okeanos c[], int n);
void printArray(Okeanos c[], int n);
int main(){
    FILE *f;
    f = fopen("project.csv","r+");
    if (f == NULL) {exit(1);}

    int count=FileOpener(f);

    f = fopen("project.csv","r+");

    Okeanos OkeanosArray[count];
    int i=0;
    int mhnas, mera, xronos;
    mhnas=mera=xronos=0;
    float thermokrasia, phosphate, silicate, nitrite, nitrate, salinity, oxygen;
    thermokrasia=phosphate=silicate=nitrite=nitrate=salinity=oxygen=0.0;
    char c;
    for (c = fgetc(f); c != '\n' && c != EOF; c=fgetc(f)){};


    while (c!=EOF)
    {
        fscanf(f, "%d/%d/%d, %f, %f, %f, %f, %f, %f, %f", &mhnas, &mera, &xronos, &thermokrasia, &phosphate, &silicate, &nitrite, &nitrate, &salinity, &oxygen);
        OkeanosArray[i].mhnas=mhnas;
        OkeanosArray[i].mera=mera;
        OkeanosArray[i].xronos=xronos;
        OkeanosArray[i].thermokrasia=thermokrasia;
        OkeanosArray[i].phosphate=phosphate;
        OkeanosArray[i].silicate=silicate;
        OkeanosArray[i].nitrite=nitrite;
        OkeanosArray[i].nitrate=nitrate;
        OkeanosArray[i].salinity=salinity;
        OkeanosArray[i].oxygen=oxygen;
        c=getc(f);
        if (c == '\n') {i++;}
    }

    /*
    for(i = 0; i<count; i++){
        printf("%d / %d / %d \n", OkeanosArray[i].mhnas , OkeanosArray[i].mera , OkeanosArray[i].xronos);

    } */

  insertionSort(OkeanosArray,count);
  printArray(OkeanosArray,count);



}


int FileOpener (FILE* f)
{
int counter = 0;
```

```
C:\pppppap\insertion_sort_1TELIKO.exe
Date: 7/21/2013 ||thermokrasia: 10.630000
Date: 11/9/2014 ||thermokrasia: 10.630000
Date: 1/25/2010 ||thermokrasia: 10.640000
Date: 7/15/2016 ||thermokrasia: 10.640000
Date: 4/5/2003 ||thermokrasia: 10.650000
Date: 1/8/2009 ||thermokrasia: 10.650000
Date: 1/10/2016 ||thermokrasia: 10.650000
Date: 2/3/2002 ||thermokrasia: 10.660000
Date: 11/18/2002 ||thermokrasia: 10.660000
Date: 11/2/2010 ||thermokrasia: 10.660000
Date: 4/11/2014 ||thermokrasia: 10.660000
Date: 4/6/2007 ||thermokrasia: 10.670000
Date: 7/21/2009 ||thermokrasia: 10.670000
Date: 1/16/2016 ||thermokrasia: 10.670000
Date: 7/19/2014 ||thermokrasia: 10.680000
Date: 4/23/2005 ||thermokrasia: 10.690000
Date: 1/13/2004 ||thermokrasia: 10.710000
Date: 11/23/2017 ||thermokrasia: 10.710000
Date: 7/23/2016 ||thermokrasia: 10.720000
Date: 7/14/2002 ||thermokrasia: 10.730000
Date: 11/15/2019 ||thermokrasia: 10.730000
Date: 4/22/2003 ||thermokrasia: 10.740000
Date: 1/19/2015 ||thermokrasia: 10.740000
Date: 1/14/2001 ||thermokrasia: 10.750000
Date: 7/24/2003 ||thermokrasia: 10.750000
Date: 3/28/2008 ||thermokrasia: 10.750000
Date: 7/13/2015 ||thermokrasia: 10.750000
Date: 1/15/2009 ||thermokrasia: 10.760000
Date: 2/1/2014 ||thermokrasia: 10.760000
Date: 11/17/2016 ||thermokrasia: 10.760000
```

Δεύτερο μέρος εκτύπωσης του πίνακα:

```
T:\sda\insertion_sort_1TELIKO.exe
Date: 4/16/2018 ||thermokrasia: 23.690001
Date: 4/18/2018 ||thermokrasia: 23.950001
Date: 4/17/2018 ||thermokrasia: 23.950001
Date: 4/11/2019 ||thermokrasia: 23.950001
Date: 4/2/2019 ||thermokrasia: 24.080000
Date: 4/3/2019 ||thermokrasia: 24.080000
Date: 4/3/2019 ||thermokrasia: 24.080000
Date: 2/12/2019 ||thermokrasia: 24.590000
Date: 10/23/2018 ||thermokrasia: 24.610001
Date: 6/10/2018 ||thermokrasia: 24.920000
Date: 2/9/2019 ||thermokrasia: 24.940001
Date: 6/12/2018 ||thermokrasia: 25.180000
Date: 6/9/2018 ||thermokrasia: 25.180000
Date: 2/10/2019 ||thermokrasia: 25.260000
Date: 2/11/2019 ||thermokrasia: 25.270000
Date: 10/16/2018 ||thermokrasia: 25.330000
Date: 10/17/2018 ||thermokrasia: 25.330000
Date: 10/29/2018 ||thermokrasia: 25.410000
Date: 10/24/2018 ||thermokrasia: 25.410000
Date: 4/14/2019 ||thermokrasia: 25.410000
Date: 4/8/2019 ||thermokrasia: 25.410000
Date: 10/26/2018 ||thermokrasia: 25.420000
Date: 10/25/2018 ||thermokrasia: 25.420000
Date: 4/13/2019 ||thermokrasia: 25.420000
Date: 4/10/2019 ||thermokrasia: 25.420000


--------------------------------
Process exited after 0.4273 seconds with return value 10
Press any key to continue . . .
```

Quicksort

```
typedef struct Okeanos
{
    int mhnas, mera, xronos;
    float  thermokrasia, phosphate, silicate, nitrite, nitrate, salinity,  oxygen;

}Okeanos;
```

```
int FileOpener (FILE* f)
{
int counter = 0;
char c;
for (c = fgetc(f); c != '\n' && c != EOF; c=fgetc(f)){};
for (c=fgetc(f); c!=EOF; c=fgetc(f))
{
    if (c == '\n')
    {
        counter++;
    }
}
    counter++;
    return counter;
}
```

Οπως εξηγησαμε και πριν εχουμε την file opener.

Συνεχιζοντας,δημιουργουμε την συναρτηση quicksort με ορισματα τον πινακα τυπου ωκεανος.

```
//QUICK SORT
void swap(int* a, int* b)
{
    int t = *a;
    *a = *b;
    *b = t;
}


int partition (Okeanos c[], int low, int high)
{
    float pivot = c[high]. thermokrasia; // pivot
    int i = (low - 1); // Index of smaller element and indicates the right position of pivot found so far
    int j;
    for ( j = low; j <= high - 1; j++)
    {
        // If current element is smaller than the pivot
        if (c[j]. thermokrasia < pivot)
        {
            i++; // increment index of smaller element
            swap(&c[i]. thermokrasia, &c[j]. thermokrasia);
            swap(&c[i].mera,&c[j].mera);
            swap(&c[i].mhnas,&c[j].mhnas);
            swap(&c[i].xronos,&c[j].xronos);
        }
    }
    swap(&c[i+1]. thermokrasia, &c[high]. thermokrasia);
    swap(&c[i+1].mera,&c[high].mera);
    swap(&c[i+1].mhnas,&c[high].mhnas);
    swap(&c[i+1].xronos,&c[high].xronos);
    return (i + 1);
}


void quickSort(Okeanos c[], int low, int high)
{
    if (low < high)
    {
      int pi = partition(c, low, high);


        quickSort(c, low, pi - 1);
        quickSort(c, pi + 1, high);
    }
}
```

1. Ουσιαστικα η quicksort ταξινομει εναν πινακα με αυξουσα σειρα με τον εξης τροπο: Επιλέγουμε ένα στοιχείο p (το οποίο ονομάζουμε pivot) και το αφαιρούμε από την πίνακα/λίστα εισόδου.
2. Χωρίζουμε τον πίνακα/λίστα σε 2 μέρη: S1 και S2, όπου το S1 θα περιέχει όλα τα στοιχεία που είναι μικρότερα από το p και το S2 όπου περιέχονται όλα τα υπόλοιπα στοιχεία τα οποία είναι μεγαλύτερα ή ίσα με p.
3. Καλούμε τον αλγόριθμο αναδρομικά στο S1, παίρνουμε απάντηση στο T1 και στο S2 και παίρνουμε απάντηση στο T2.
4. Επιστρέφουμε τον πίνακα [T1, p, T2]

Βαζουμε τον πινακα Okeanos c[] και τον χωριζουμε σε δυο κομματια με την χρηση του pivot το οποιο εχουμε ορισει εμεις να ειναι η θερμοκρασια.που εχουμε ορισει.Αρα εχουμε το κομματι low με στοιχεια μικροτερα απο το pivot και το high κομματι με στοιχεια μεγαλυτερα απο το pivot.

Χρησιμουποιουμε δυο μετρητες i και j οι οποιοι διατρεχουν τον πινακα δεξια προς αριστερα και αριστερα προς δεξια αντιστοιχα.Ο μετρητης i ψαχνει για στοιχεια μικροτερα του pivot και ο j για στοιχεια μεγαλυτερα απο το pivot.Οταν βρεθουν και τα δυο στοιχεια γινεται swap των τιμων i και j .Αυτη η διαδικασια γινεται ωσπου ο πινακας να γινει sorted(δηλαδη ολα τα στοιχεια αρσιτερα του pivot να ειναι μικροτερα και ολα τα στοιχεια δεξια μεγαλυτερα.)Στην συνεχεια χωριζουμε τον πινακα (γινεται partition) και επαναλαμβανουμε την διαδικασια μεχρι να ειναι ολα τα στοιχεια ταξινομημενα με αυξουσα σειρα.

Καλουμε στην main την συνάρτηση μας QuickSort με ορίσματα τον πίνακα μας OkeanosArray που περιέχει τα στοιχεία του αρχείου ,την αρχή και το τέλος του πίνακα. Μετα χρησιμοποιουμε μια συναρτηση(την printarray)για να εκτυπωσουμε τα αποτελεσματα του πινακα.

```
/* Function to print an array */
void printArray(OCEAN c[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        printf("Date: %d/%d/%d ||Temp: %f \n",c[i].month,c[i].day,c[i].year,c[i].temp);
    printf("\n");
}
```

```c
int main(){
    FILE *fp;
    fp = fopen("project.csv","r+");
    if (fp == NULL) {exit(1);}

    int count=FileCounter(fp);

    fp = fopen("project.csv","r+");

    OCEAN OceanArray[count];
    int i=0;
    int month, day, year;
    month=day=year=0;
    float temp, phos, silic, nitrite, nitrate, sal, oxyg;
    temp=phos=silic=nitrite=nitrate=sal=oxyg=0.0;
    char c;
    for (c = fgetc(fp); c != '\n' && c != EOF; c=fgetc(fp)){};


    while (c!=EOF)
    {
        fscanf(fp, "%d/%d/%d, %f, %f, %f, %f, %f, %f, %f", &month, &day, &year, &temp, &phos, &silic, &nitrite, &nitrate, &sal, &oxyg);
        OceanArray[i].month=month;
        OceanArray[i].day=day;
        OceanArray[i].year=year;
        OceanArray[i].temp=temp;
        OceanArray[i].phos=phos;
        OceanArray[i].silic=silic;
        OceanArray[i].nitrite=nitrite;
        OceanArray[i].nitrate=nitrate;
        OceanArray[i].sal=sal;
        OceanArray[i].oxyg=oxyg;
        c=getc(fp);
        if (c == '\n') {i++;}
    }

    /*
    for(i = 0; i<count; i++){
        printf("%d / %d / %d \n", OceanArray[i].month , OceanArray[i].day , OceanArray[i].year);

    } */

  quickSort(OceanArray,0,count);
  printArray(OceanArray,count);
```

Αποτελεσματα εκτυπωσης του πινακα :

```
Select T:\sda\quicksort1TELIKO.exe
Date: 4/16/2018 || thermokrasia: 23.650000
Date: 4/16/2018 || thermokrasia: 23.690001
Date: 4/11/2019 || thermokrasia: 23.950001
Date: 4/18/2018 || thermokrasia: 23.950001
Date: 4/17/2018 || thermokrasia: 23.950001
Date: 4/3/2019 || thermokrasia: 24.080000
Date: 4/3/2019 || thermokrasia: 24.080000
Date: 4/2/2019 || thermokrasia: 24.080000
Date: 2/12/2019 || thermokrasia: 24.590000
Date: 10/23/2018 || thermokrasia: 24.610001
Date: 6/10/2018 || thermokrasia: 24.920000
Date: 2/9/2019 || thermokrasia: 24.940001
Date: 6/9/2018 || thermokrasia: 25.180000
Date: 6/12/2018 || thermokrasia: 25.180000
Date: 2/10/2019 || thermokrasia: 25.260000
Date: 2/11/2019 || thermokrasia: 25.270000
Date: 10/17/2018 || thermokrasia: 25.330000
Date: 10/16/2018 || thermokrasia: 25.330000
Date: 10/29/2018 || thermokrasia: 25.410000
Date: 4/14/2019 || thermokrasia: 25.410000
Date: 4/8/2019 || thermokrasia: 25.410000
Date: 10/24/2018 || thermokrasia: 25.410000
Date: 4/13/2019 || thermokrasia: 25.420000
Date: 10/25/2018 || thermokrasia: 25.420000
Date: 10/26/2018 || thermokrasia: 25.420000


--------------------------------
Process exited after 0.4624 seconds with return value 10
Press any key to continue . . .
```

Counting Sort:

Οπως εχουμε εξηγησει και στις προημουμενες συναρτησεις εχουμε το file opener και τα γνωστα ορισματα.

Η counting sort μετραει αρχικα ποσες φορες εμφανιζεται η καθε τιμη που υπαρχει στον πινακα και στην συνεχεια αναθετει σε καθε τιμη ενα ξεχωριστο key (ειδος hashing) ωστε να τοποθετηθουν με την σωστη σειρα (αυξουσα).

Εφαρμοζουμε τον αλγοριθμο στον πινακα Okeanos c[] με βαση την τιμη phosphate.Ο παρακάτω αλγόριθμος ωστόσο δουλεύει αποκλειστικά για ακέραιες τιμές επομενως (όπως φαινεται στις παρακάτω εικόνες) πολλαπλασιάζουμε με το 100 για να κάνουμε τις float τιμές μας ακέραιες , τις οποίες αποθηκεύουμε σε μια νεα λίστα Counting List με μεγεθος ίσο με το OkeanosArray και πλήρως ακέραιες τιμες με σκοπό την ορθή λειτουργια της συανρτησης.

```
    int CountingList[count];
    int j;

    for(j=0; j<count; j++){
        CountingList[j]=OkeanosArray[j].phosphate*100;
    }
    /*
    for(j=0; j<count; j++){
        printf("%d\n",CountingList[j]);
    }
    */

   //int length = sizeof(CountingList) / sizeof(CountingList[0]);

    countingSort(CountingList,OkeanosArray,count);

}
```

Μετα με την συναρτηση printArray εκτυπωνουμε τα αποτελεσματα του πινακα

```c
/* Function to print an array */
void printArray(Okeanos c[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        printf("Date: %d/%d/%d ||phosphate: %f \n",c[i].mhnas,c[i].mera,c[i].xronos,c[i].phosphate);
    printf("\n");
}
```

Τα αποτελεσματα εκτυπωμενα :

```c
    while (c!=EOF)
    {
        fscanf(fp, "%d/%d/%d, %f, %f, %f, %f, %f, %f, %f", &mhnas, &mera, &xronos, &thermokrasia, &phosphate, &silicate, &nitrite, &nitrate, &salinity, &oxygen);
        OkeanosArray[i].mhnas=mhnas;
        OkeanosArray[i].mera=mera;
        OkeanosArray[i].xronos=xronos;
        OkeanosArray[i].thermokrasia=thermokrasia;
        OkeanosArray[i].phosphate=phosphate;
        OkeanosArray[i].silicate=silicate;
        OkeanosArray[i].nitrite=nitrite;
        OkeanosArray[i].nitrate=nitrate;
        OkeanosArray[i].salinity=salinity;
        OkeanosArray[i].oxygen=oxygen;
        c=getc(fp);
        if (c == '\n') {i++;}
    }



    int CountingList[count];
    int j;

    for(j=0; j<count; j++){
        CountingList[j]=OkeanosArray[j].phosphate*100;
    }
    /*
    for(j=0; j<count; j++){
        printf("%d\n",CountingList[j]);
    }
    */

    //int length = sizeof(Countinglist) / sizeof(CountingList[0]);

    countingSort(CountingList,OkeanosArray,count);

}
```

```
Select C:\ppppap\counting_sort.exe
Date: 6/23/2018 ||phosphate: 271
Date: 4/16/2018 ||phosphate: 273
Date: 4/8/2018 ||phosphate: 273
Date: 6/13/2018 ||phosphate: 274
Date: 4/10/2013 ||phosphate: 276
Date: 4/16/2014 ||phosphate: 277
Date: 11/5/2015 ||phosphate: 279
Date: 7/26/2009 ||phosphate: 280
Date: 8/4/2010 ||phosphate: 281
Date: 7/19/2009 ||phosphate: 287
Date: 11/5/2006 ||phosphate: 288
Date: 2/5/2018 ||phosphate: 291
Date: 8/2/2010 ||phosphate: 295
Date: 1/19/2008 ||phosphate: 301
Date: 11/14/2005 ||phosphate: 302
Date: 11/21/2005 ||phosphate: 314
Date: 11/11/2005 ||phosphate: 322
Date: 11/12/2005 ||phosphate: 326
Date: 2/2/2015 ||phosphate: 332


--------------------------------
Process exited after 0.7027 seconds with return value 10
Press any key to continue . . .
```

Heapsort

Ο αλγοριθμος heapsort εισαγει ολες τις τιμες σε ενα δυαδικο δενδρο και συγκρινει καθε φορα τον γονεα με το παιδι.Εαν το παιδι εχει μεγαλυτερη τιμη απο τον γονεα τοτε αλλαζουν θεση (swap).Οταν γινει αυτη η διαδικασια αφαιρουμε το μεγαλυτερο στοιχειο και το εισαγουμε στον πινακα.Χρησιμοποιούμε την βοηθητικη συνάρτηση heapify με ορίσματα τον πίνακα, το μέγεθός του και το στοιχείο που γιενται η συγκριση μεταξύ γονεα και παιδιού.

Οπως εχουμε εξηγησει και στις προημουμενες συναρτησεις εχουμε το file opener και τα γνωστα ορισματα.

Εφαρμοζουμε την συναρτηση στον πινακα Okeanos c[] με βαση την τιμη phosphate.

```
}

void heapify(Okeanos c[], int n, int i) {
  int max = i; //Initialize max as root
  int leftChild = 2 * i + 1;
  int rightChild = 2 * i + 2;

  //If left child is greater than root
  if (leftChild < n && c[leftChild].phosphate > c[max].phosphate)
    max = leftChild;

  //If right child is greater than max
  if (rightChild < n && c[rightChild].phosphate > c[max].phosphate)
    max = rightChild;

  //If max is not root
  if (max != i) {
    swap(&c[i].phosphate, &c[max].phosphate);
    swap(&c[i].mera,&c[max].mera);
    swap(&c[i].mhnas,&c[max].mhnas);
    swap(&c[i].xronos,&c[max].xronos);
    //heapify the affected sub-tree recursively
    heapify(c, n, max);
  }
}
```

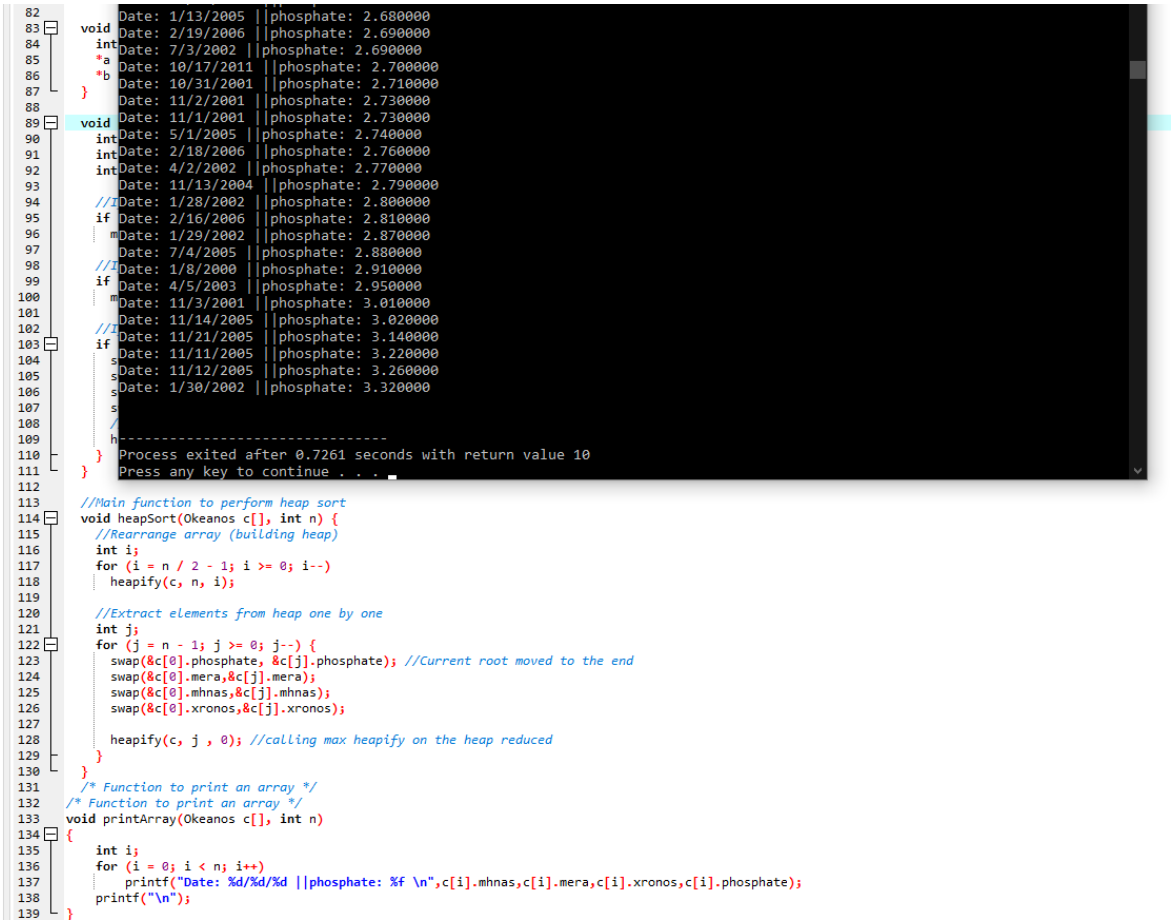
```
//Main function to perform heap sort
void heapSort(Okeanos c[], int n) {
  //Rearrange array (building heap)
  int i;
  for (i = n / 2 - 1; i >= 0; i--)
    heapify(c, n, i);

  //Extract elements from heap one by one
  int j;
  for (j = n - 1; j >= 0; j--) {
    swap(&c[0].phosphate, &c[j].phosphate); //Current root moved to the end
    swap(&c[0].mera,&c[j].mera);
    swap(&c[0].mhnas,&c[j].mhnas);
    swap(&c[0].xronos,&c[j].xronos);

    heapify(c, j , 0); //calling max heapify on the heap reduced
  }
}
 /* Function to print an array */
/* Function to print an array */
```

Αρα κανουμε στην ουσια heap με την τιμη phosphate ενω παραλληλα θα αναγραφεται και γίνεται swap μέσω της αντιστοιχη συναρτησης η μερα,ο μηνας και ο χρονος που αντιστοιχουν σε αυτη την τιμη.

Μετα με την συναρτηση printArray εκτυπωνουμε τα αποτελεσματα του πινακα στην main αφου εχουμε χρησιμοποιήσει τις κατάλληλες συναρτήσεις και ορίσματα.

```
/* Function to print an array */
void printArray(Okeanos c[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        printf("Date: %d/%d/%d ||phosphate: %f \n",c[i].mhnas,c[i].mera,c[i].xronos,c[i].phosphate);
    printf("\n");
}
```

Τα εκτυπωμενα αποτελεσματα :



Binary search

Οπως εχουμε εξηγησει και στις προημουμενες συναρτησεις εχουμε το file opener και τα γνωστα ορισματα.

Η binary search συγκρινει την τιμη της μεσης του πινακα με αυτη που ψαχνουμε.Αν η τιμη της μεσης ειναι μικροτερη απο αυτη που ψαχνουμε τοτε συνεχιζουμε τον αλγοριθμο στην δεξια μερια του πινακα.Εαν ειναι μεγαλυτερη συνεχιζουμε στο αριστερο μερος του πινακα.Οταν βρουμε την τιμη που ψαχνουμε τοτε ο αλγοριθμος σταματαει.

Την μεση την βρισκουμε με τον τυπο : (Αρχη+τελος)/2

Εφαρμοζουμε την συναρτηση στον πινακα OkeanosArray[] με βαση τις τιμες μερα,μηνας και χρονος.

```
int binarySearch(Okeanos array[], int mera,int mhnas,int xronos, int low, int high) {

    while(low<=high) {
    int mid =low+ (high - low) / 2;

   if ((array[mid].xronos<xronos)){
        low=mid+1;
        return binarySearch(array, mera,mhnas,xronos, low, high);
    }
    else if((array[mid].xronos>xronos)){
        high=mid-1;
        return binarySearch(array,mera,mhnas,xronos, low, high);
     }else if(array[mid].xronos==xronos){
        if(array[mid].mhnas<mhnas){
            low=mid+1;
            return binarySearch(array, mera,mhnas,xronos, low, high);
        }
        else if(array[mid].mhnas>mhnas){
            high=mid-1;
            return binarySearch(array,mera,mhnas,xronos, low, high);
        }
        else if(array[mid].mhnas==mhnas){
            if(array[mid].mera<mera){
                low=mid+1;
                return binarySearch(array, mera,mhnas,xronos, low, high);
            }
            else if(array[mid].mera>mera){
                high=mid-1;
                return binarySearch(array,mera,mhnas,xronos, low, high);
            }
            else if((array[mid].mera == mera)){
                return array[mid].thermokrasia;
    }
        }
    }

    }
  return -1;
}
```

Δημιουργούμε συναρτηση userInput η οποια ζηταει απο τον χρηστη να εισαγει μια τιμη (μερα,μηνας , χρονος).Η συνάρτηση αυτή παίρνει ως ορίσματα το string s το οποίο εκτυπώνει στον χρήστη το κατάλληλο μήνυμα για το τι θα εισάγει (μήνα,μέρα , χρόνο),τον δείκτη Value ο οποίος περνά στο περιεχόμενο της διεύθυνσης του στοιχείου που έχουμε στην main την ημερομηνία που έχει εισάγει ο χρήστης στην συνάρτηση και τα ορια της επιτρεπόμενης τιμής που μπορεί να εισάγει ο χρηστης (1 ως 31 για μερες) (1 ως 12 μηνες) (2000ως 2019 για χρόνια).Σε περίπτωση που ο χρήστης εισάγει διαφορετική τιμή από τα επιτρεπόμενα όρια, μέσω της do while εκτυπώνεται κατάλληλο μήνυμα μέχρι να εισαχθει αποδεχτη ημερομηνια.

```c
void userInput(char s[] ,int *Value,int min ,int max ){

    int TestValue;

    do{

      printf("Enter the %s :",s);

      scanf("%d",&TestValue);

      if(TestValue<min|| TestValue>max){

          printf("Error %s must be between %d-%d",s,min,max);

          TestValue=0;

          *Value=TestValue;

          printf("\n");

      }
      else{
          printf("You entered :%d",TestValue);
          *Value=TestValue;
          printf("\n");


      }
    }while(*Value<min || *Value>max);


}
```

Στην main:

```c
int Uday,Umonth,Uyear;
  userInput("Day",&Uday,1,31);
  userInput("Month",&Umonth,1,12);
  userInput("Year",&Uyear,2000,2019);
  printf("You entered :%d/%d/%d",Uday,Umonth,Uyear);
  int index,index2;
```

Επιπλεον , όπως προαναφέραμε η binary search λειτουργεί αποκλειστκα σε ταξινομημένους πίνακες.Χρησιμοποιούμε λοιπον την παρακάτω

συνάρτηση sort η οποία ταξινομεί τον πίνακα κατά αύξουσα σειρα:

```
void Sort(OCEAN d[],int n){
 int i,k,j;
for(i=0; i<n; i++){
for(j=0; j<=i; j++){

if(d[j].year>d[i].year){
k=d[j].year;
d[j].year=d[i].year;
d[i].year=k;
k=d[j].month;
d[j].month=d[i].month;
d[i].month=k;
k=d[j].day;
d[j].day=d[i].day;
d[i].day=k;
}
else
{

if(d[j].year==d[i].year){

if(d[j].month>d[i].month){
k=d[j].year;
d[j].year=d[i].year;
d[i].year=k;
k=d[j].month;
d[j].month=d[i].month;
d[i].month=k;
k=d[j].day;
d[j].day=d[i].day;
d[i].day=k;
}
```

```
else
{
if(d[j].month==d[i].month)
{
    if(d[j].day>d[i].day)
    {
    k=d[j].year;
    d[j].year=d[i].year;
    d[i].year=k;
    k=d[j].month;
    d[j].month=d[i].month;
    d[i].month=k;
    k=d[j].day;
    d[j].day=d[i].day;
    d[i].day=k;
    }
}

}
}
}

}

}
for(i=0; i<n; i++)
{
printf("\n%d/%d/%d",d[i].month,d[i].day,d[i].year);
}
}
```

Στη συνεχεια με τον αλγοριθμο binary search, ο οποίος εξηγήθηκε και ψαχνουμε που ειναι η τιμη που ζητηθηκε. Αν βρεθει μας εμφανιζει την θερμοκρασια της ημερομηνιας που εβαλε .Σε περιπτωση δεν υπαρχει εμφανιζει το αναλογο μηνυμα.

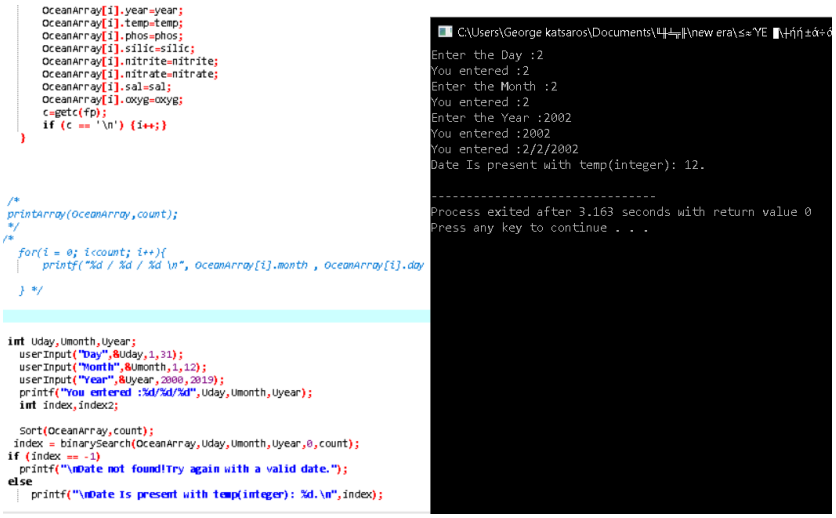
Μετα με την συναρτηση printArray εκτυπωνουμε τα αποτελεσματα του πινακα που ζητησε ο χρηστης.

```
/* Function to print an array */
void printArray(Okeanos c[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        printf("Date: %d/%d/%d  \n",c[i].mhnas,c[i].mera,c[i].xronos);
    printf("\n");
}
```

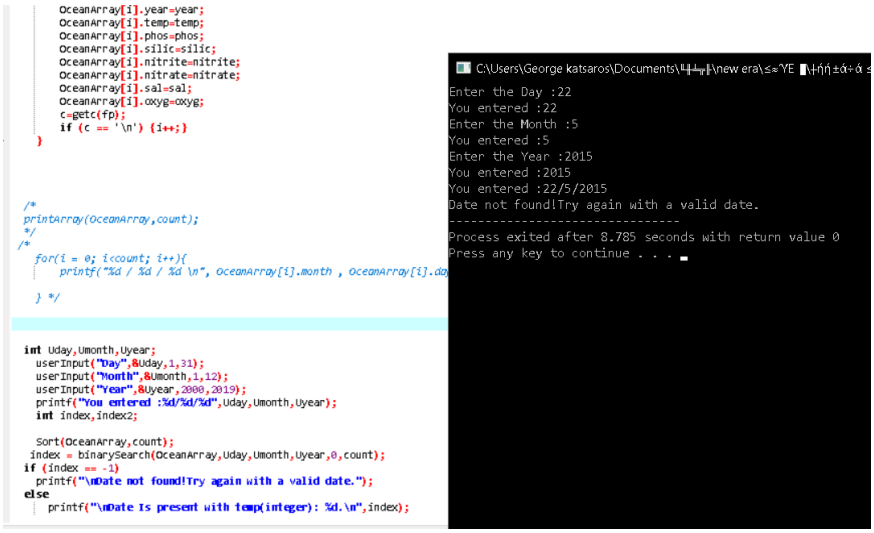
Τέλος στην main χρησιμοποιουμε την ακεραια μεταβλητή index η οποία κρατάει την τιμή που επιστρέφη η binarysearch και εκτυπωνει το κατάλληλο μηνυμα σε περιπτωση που η ημερομηνία υπάρχει ή δεν υπάρχει.Η ίδια μεταβλητή χρησιμοποιείται σε όλα τα παρακάτω ερωτήματα.

Τα εκτυπωμενα αποτελεσματα :

Α)Για τιμή που υπάρχει στον πίνακα



Β)Για τιμή που δεν υπάρχει στον πίνακα:



Interpolation Search

Προκειμένου να υλοποιήσουμε τις παρακάτω συναρτήσεις (interpolation και bis) αλλάζουμε την δομή του προγραμματος στα παρακάτω σημεία.

Αρχικά πλεον αλλάζουμε την δομή μας σε ονομα Data με σκοπό την διαφορετική αναγνωση της ημερομηνιας από το προγραμμα.Χρησιμοποιουμε την ακεραια μεταβλητη long int x η οποια αποθηκεύει την ημερομηνια ως ένα ενιαίο συνολο με την μορφή(YYYYMMDD).Οι float μεταβλητές μας παραμένουν αμετάβλητες. Αυτό το κάνουμε διοτι πλεον επεξεργαζόμαστε,διαβάζουμε και χρησιμοποιούμε την ημερομηνία(data/Month/Date) ως ενα στοιχείο(με όνομα x) το οποίο παιρνει την μορφή YYYYMMDD.

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <string.h>
#include <time.h>


typedef struct Data
{
    long int x;
    float temp, phos, silic, nitrite, nitrate, sal, oxyg;
}Data;
```

Επίσης επεξεργαζόμαστε κατάλληλα όλες τις συναρτήσεις και τα ορίσματά μας , βασισμένα στις παρακάτω αλλαγες για αυτό το ερώτημα.

Χρησιμοποιούμε τις συναρτήσεις Readfile ,xSwap και RemoveChar(οι οποίες κανουν την δουελια της παραπάνω file opener) με τα κατάλληλα

ορίσματα,μεταβλητές και δείκτες.Μέσω αυτών αλλάζουμε τον τρόπο με τον οποίο διαβάζεται το αρχειο και πλεον περνάμε την ημερομηνία ως ένα στοιχείο(το ονομαζουμε x στην δομή μας) και αποθηκεύουμε εκεί την ημερομηνία με σκοπό την ορθή διαχείρηση της στην συνάρτηση μας.Ο τρόπος που γινεται αυτό είναι μέσω των συναρτησεων xSwap και RemoveChar οι οποίες κατά την ανάγνωσητου αρχείου Readfile παραλείπουν τα κομματα και το συμβολο / και ετσι μπορει να διαβασθεί ορθα η ημερομηνια(μορφη YYYYMMDD).

```
FILE* fp = fopen("project.csv", "r");
int rows = -1;
if (!fp){printf("Can't open file\n");}
else
{
    char buffer[1024];
    int r=0;
    while (fgets(buffer,1024, fp)){
        rows++;
        if (rows == 1){continue;}
        char* value = strtok(buffer, ", ");
        while (value)
        {value = strtok(NULL, ", ");}
    }
}
fclose(fp);
```

```
void ReadFile(Data* DataArray)
{
    FILE* fp = fopen("project.csv", "r");
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int row = 0;
        int column = 0;
        int r=-1;
        while (fgets(buffer,1024, fp)){
            column = 0;
            row++;
            r++;
            if (row == 1)
            {
                r--;
                continue;
            }

            char* value = strtok(buffer, ", ");
            while (value) {
                if (column == 0)
                {
                    RemoveChar(value, '/');
                    xSwap(value);
                    DataArray[r].x = atol(value);
                }
                if (column == 1){DataArray[r].temp = atof(value);}
                if (column == 2){DataArray[r].phos = atof(value);}
                if (column == 3){DataArray[r].silic = atof(value);}
                if (column == 4){DataArray[r].nitrite = atof(value);}
                if (column == 5){DataArray[r].nitrate = atof(value);}
                if (column == 6){DataArray[r].sal = atof(value);}
                if (column == 7){DataArray[r].oxyg = atof(value);}
                value = strtok(NULL, ", ");
                column++;
            }
        }
        fclose(fp);
    }
```

```
void RemoveChar(char* s, char c)
{
    int i, j;
    int len = strlen(s);
    for(i=0; i<len; i++)
    {
        if(s[i] == c)
        {
            for(j=i; j<len; j++){s[j] = s[j+1];}
            len--;
            i--;
        }
    }
}

void xSwap(char* str){
    char temp;
    int n;

    n = strlen(str);
    int i;
    for ( i=0; i<4; i++){
        temp = str[i];
        str[i] = str[n-(4-i)];
        str[n-(4-i)] = temp;
    }
}
```

Επιπλέον τροποποιούμε κατάλληλα ξανα την UserInput με σκοπό την ορθή λειτουργεία της.Η διαδικασία και τα ορίσματα είναι αντίστοιχα με πριν ωστόσο πλεον δεν είναι απαραιτητη η μεταβλητη char s που εκτύπωνε το μηνυμα στον χρήστη για το αν θα πληκτρολογησει μηνα, μερα ή χρόνο διότι πλεον πληκτρολογεί μια ενιαια ημερομηνία με το κατάλληλο μήνυμα.Επίσης αλλάζουμε τα όρια σε long και σε μορφή(YYYYMMDD).Τα νεα μας οριa είναι 00000000 ως 20191231.

```
void userInput(long int *Value,long int min ,long int max ){
    long int TestValue;
    do{
      printf("Please select a desired x in YYYYMMDD format:\n");
    scanf("%ld",&TestValue);
      if(TestValue<min|| TestValue>max){
          printf("Error x must be between %ld-%ld",min,max);
          TestValue=0;
          *Value=TestValue;
          printf("\n");

      }
      else{
          printf("You entered :%ld",TestValue);
          *Value=TestValue;
          printf("\n");


      }
    }while(*Value<min || *Value>max);

}
```

Πλεον αφου γινει η αναγνωση του αρχειου δημιουργουμε τον πινακα τυπου data με ονομα DataArray και στοιχεια ίσα με rows τα οποια εχουν μετρηθει με βαση το αρχειο.Στον πίνακα αυτό περνάμε τις τιμές μας (ημερομηνια και float στοιχεία).

Όλη η παραπάνω διαδικασία γίνεται με σκοπο την λειτουργία της συνάρτησης InterpolationSearch. Στη αναζήτηση παρεμβολής (interpolation search) η εύρεση της εγγραφής στον (ταξινομημένο) πίνακα γίνεται υπολογίζοντας την πιθανή θέση της εγγραφής σε σχέση

με τις τιμές των στοιχείων στα άκρα της περιοχής αναζήτησης. Ανάλογα με τη σύγκριση της τιμής της εγραφής με την εγγραφή στην πιθανή θέση αναπροσαρμόζεται το αντίστοιχο άκρο.Αν ο πίνακας έχει στοιχεία ομοιόμορφα κατανεμημένα θα απαιτηθούν O(log log n) συγκρίσεις.Στην ουσία εξετάζοντας πόσο μεγαλύτερο είναι το ζητούμενο στοιχείο από το στοιχείο του αριστερού άκρου, καθώς και πόσο μεγαλύτερο είναι το στοιχείο του δεξιού από το στοιχείο του αριστερού άκρου, γίνεται εκτίμηση της θέσης του x λαμβάνοντας το λόγο των δύο διαφορών.

Στον κώδικά μας χρησιμοποιούμε ως ορίσματα τον πινακα d τύπου Data, την αρχή, το τέλος του και το στοιχειο μορφής(yyyymmdd) που θελουμε να βρούμε.Εκτελούμε την υλοποίηση τουαλγορίθμου με βάση τις σημειώσεις του βιβλίου E_BOOK ΒΙΒΛΙΟΥ ΔΟΜΩΝ ΔΕΔΟΜΕΝΩΝ ΟΜΟΤΙΜΟΥ ΚΑΘΗΓΗΤΗ Α. ΤΣΑΚΑΛΙΔΗ:

```
float interpolation(Data array[],int data, int low, int high)
{
    int pos;

    if (low <= high && data >= array[low].x && data <= array[high].x) {

        pos = low + (((double)(high - low) / (array[high].x - array[low].x))* (data - array[low].x));

    if ((array[pos].x<data)){
        low= pos+1;
        return interpolation(array,data, low, high);
    }
    else if((array[pos].x>data)){
        high=pos-1;
        return interpolation(array,data, low, high);
     }else if(array[ pos].x==data){

                return array[pos].temp;
    }
}

    return -1;


}
```

Τέλος καλούμε στην main τις παραπάνω συναρτήσεις για την εισαγωγή τιμης από τον χρήστη, την κατάλληλη ανάγνωση του αρχείου, την δημιουργία πινακα δομής τύπου Data και τις συναρτησεις interpolation

και printarray που εξηγηθηκαν.

```c
int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);

    long int input;
    userInput(&input,20000101,20191231);
    float result=interpolation(DataArray,input,0,rows-1);
    if(result ==-1){
        printf("Date does not exist.Try again with a valid date.");
    }
    else{
        printf("Temp for the current date is: %.2f",result);
    }


}
```

Παραθέτουμε τις εκτυπώσεις αφού τρέξουμε τον κωδικα:

Α) Για μη επιτρεπτη και υστερα επιτρεπτή ημερομηνια που υπαρχει στον πίνακα:

```c
void Printday(Data* day, int measure);
void ReadFile(Data* Oceanday);
void userInput(long int *value,long int min ,long int max );
float interpolation(Data array[],int data, int low, int high);

int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);

    long int input;
    userInput(&input,20000101,20191231);
    float result=interpolation(DataArray,input,0,rows-1);
    if(result ==-1){
        printf("Date does not exist.Try again with a valid date.");
    }
    else{
        printf("Temp for the current date is: %.2f",result);
    }


}
```

```
C:\Users\George katsaros\Documents\...\new era\...
Please select a desired x in YYYYMMDD format:
31233344
Error x must be between 20000101-20191231
Please select a desired x in YYYYMMDD format:
20020406
You entered :20020406
Temp for the current date is: 10.59
--------------------------------
Process exited after 18.9 seconds with return value 35
Press any key to continue . . .
```

B) Για μη επιτρεπτη και υστερα επιτρεπτή ημερομηνια που δεν υπαρχει στον πίνακα:

```c
void Printday(Data* day, int measure);
void ReadFile(Data* Oceanday);
void userInput(long int *Value,long int min ,long int max );
float interpolation(Data array[],int data, int low, int high);

int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);

    long int input;
    userInput(&input,20000101,20191231);
    float result=interpolation(DataArray,input,0,rows-1);
    if(result ==-1){
        printf("Date does not exist.Try again with a valid date.");
    }
    else{
        printf("Temp for the current date is: %.2f",result);
    }


}
```

```
C:\Users\George katsaros\Documents\╚╫╤╥╟\new era\≤≈ΎΕ █\┼ήή±ά÷ά
Please select a desired x in YYYYMMDD format:
45678901
Error x must be between 20000101-20191231
Please select a desired x in YYYYMMDD format:
20151211
You entered :20151211
Date does not exist.Try again with a valid date.
--------------------------------
Process exited after 13.52 seconds with return value 48
Press any key to continue . . .
```

Binary Interpolation Search:

Η παρακάτω συνάρτηση υλοποιείται με βάση τις αλλαγές που προαναφέραμε στην interpolation search.

Συνοπτικά, χρησιμοποιούμε ξανα την δομη Data , τις συναρτήσεις Removechar,xswap και readfile για την ανάγνωση και την δημιουργία πίνακα δομής data , την userinput για την εισαγωγη τιμης απτον χρήστη και την printday για την εκτύπωση (βοηθητική συναρτηση).Ολες οι παραπανω δομες,δεικτες και συναρτήσεις εξηγηθηκαν.

Η bis προσπαθεί να εντοπίσει το προς αναζήτηση στοιχείο περιορίζοντας σε κάθε επανάληψη το σχετικό διάστημα(οπως η interpolation). Ο περιορισμός αυτός γίνεται με επιλογή ενός στοιχείου στο τρέχον διάστημα και με βάση αυτό το στοιχείο γίνεται ο διαχωρισμός του διαστήματος σε δύο υποδιαστήματα. Στην αναζήτηση παρεμβολής επιλέγεται το στοιχείο σύμφωνα με τον τύπο:

$$next_{\text{INT}} \leftarrow \left\lfloor \frac{key - A[left]}{A[right] - A[left]} (right - left) \right\rfloor + left$$

Ο κώδικας της bis υλοποιείται με βάση τις σημειώσεις του βιβλίου Ε_ΒΟΟΚ ΒΙΒΛΙΟΥ ΔΟΜΩΝ ΔΕΔΟΜΕΝΩΝ ΟΜΟΤΙΜΟΥ ΚΑΘΗΓΗΤΗ Α. ΤΣΑΚΑΛΙΔΗ.Χρησιμοποιούμε ως τιμές τον πίνακα δομής τύπου data με όνομα d, το στοιχείο χ το οποίο είναι το στοιχείο που αναζητάμε,την αρχή και το τέλος του πίνακα.

Ακόμη χρησιμοποιούμε την βοηθητική συνάρτηση linearsearch που εκτελεί σειριακή αναζήτηση με σκοπό την εύρεση κάποιων τιμών που δεν μπορουσαμε να εντοπίσουμε κατά την δημιουργία της bis από μόνη της.

Παραθέτουμε τις bis και linearsearch:

```c
float bis(Data d[], long int x,int right,int left){
    int measure=right-left;
    int next;
    int i;
    next=(int) measure * (x-d[left].x)/(d[right].x-d[left].x) + 1;
    while(x!=d[next].x && left<right){
        i=0;
        measure=right-left;
        if(measure <= 3){return LinearSearch(d, left, right, x);}
        if(x >= d[next].x)
        {
            while(x > d[next + i * ((int) sqrt(measure))-1].x){i++;}
            right = next + (int) (i * sqrt(measure));
            left = next + (int) ((i-1) * sqrt(measure));
        }
        else if(x < d[next].x)
        {
            while(x < d[next - i * ((int)sqrt(measure))+1].x){i++;}
            right = next - (int)((i-1) * sqrt(measure));
            left = next - (int)(i * sqrt(measure));
        }
        next = (int) (left + ((right - left + 1) * (x - d[left].x)/(d[right].x - d[left].x))) - 1;
    }
    if(x == d[next].x){
        return d[next].thermokrasia;}
    else{

        return -1;
    }
}
float LinearSearch(Data* arr, int left, int right, long int date)
{
    int i = left, found = 0;
    while (!found && i < right){
        if (arr[i].x == date){
        return arr[i].thermokrasia;
        }
        ++i;
    }
```

Επιπλέον παραθέτουμε την main και τις κατάλληλες εκτυπώσεις:

```c
void userInput(long int *Value,long int min ,long int max );
float bis(Data d[], long int x,int right,int left);
float LinearSearch(Data* arr, int left, int right, long int date);

int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);

    long int input;
    userInput(&input,20000101,20191231);
    float result=bis(DataArray,input,rows-1,0);
    if(result !=-1){
        printf("thermokrasia for the current date is: %.2f",result);
    }
    else{
        printf("Date does not exist.Try again with a valid date.");
    }
```

```
C:\Users\George katsaros\Documents\╙╫╤╢\new era\≤≈ΎΕ ▐\┼ήή±ά÷ά ≤≈ΎΕ
Please select a desired x in YYYYMMDD format:
20070303
You entered :20070303
Date does not exist.Try again with a valid date.
--------------------------------
Process exited after 4.067 seconds with return value 48
Press any key to continue . . .
```


```c
void userInput(long int *Value,long int min ,long int max );
float bis(Data d[], long int x,int right,int left);
float LinearSearch(Data* arr, int left, int right, long int date);

int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);

    long int input;
    userInput(&input,20000101,20191231);
    float result=bis(DataArray,input,rows-1,0);
    if(result !=-1){
        printf("thermokrasia for the current date is: %.2f",result);
    }
    else{
        printf("Date does not exist.Try again with a valid date.");
    }
```


```
C:\Users\George katsaros\Documents\╙╫╤╢\new era\≤≈ΎΕ ▐\┼ήή±ά÷ά ≤≈ΎΕύ≥\4Ύ ┼╫┴╟
Please select a desired x in YYYYMMDD format:
20060715
You entered :20060715
thermokrasia for the current date is: 8.23
--------------------------------
Process exited after 5.822 seconds with return value 42
Press any key to continue . . .
```

PART 2:

Για την υλοποίηση του part II δημιουργήσαμε ένα αρχείο με ονομα menu.c το οποίο εκτυπώνει στον χρήστη το κατάλληλο μηνυμα και τον αφήνει να περιηγηθεί στον χώρο του μενου χωρίς ωστόσο να έχουμε υλοποιήσει κάποιο άλλο ερώτημα.Παραθέτουμε τον κώδικα που φτιάξαμε και χρησιμοποιήσαμε από τα προηγούμενα ερωτήματα καθώς και την περιήγηση στο μενου:

```c
typedef struct Data
{
    long int x;
    float temp, phos, silic, nitrite, nitrate, sal, oxyg;
}Data;


void RemoveChar(char* str, char c);
void xSwap(char* str);
void Printday(Data* day, int measure);
void ReadFile(Data* Oceanday);
void userInput(long int *Value,long int min ,long int max );

int main()
{
    FILE* fp = fopen("project.csv", "r");
    int rows = -1;
    if (!fp){printf("Can't open file\n");}
    else
    {
        char buffer[1024];
        int r=0;
        while (fgets(buffer,1024, fp)){
            rows++;
            if (rows == 1){continue;}
            char* value = strtok(buffer, ", ");
            while (value)
            {value = strtok(NULL, ", ");}
        }
    }
    fclose(fp);
    Data DataArray[rows];
    ReadFile(DataArray);
```

```c
    Data DataArray[rows];
    ReadFile(DataArray);
    //Printday(DataArray, rows);
    int choice=NULL ;
while(choice==NULL)
{
    system("cls");
    printf("Menu Heap:");
    printf("\n==============");
    printf("\n1-Eisagwgi");
    printf("\n2-Diagrafi");
    printf("\n3-Ektypwsi");
    printf("\n4-HeapSort!");
    printf("\n5-Eksodos");
    printf("\nEpilogi?\n");
    scanf("%d",&choice);
    switch(choice){
    case 1:
      //printlist();
      break;

    case 2:
      //editStud();
      break;

    case 3:
    printf("ff");
      //deLStud();
      break;

    case 4:
      //addStud();
      break;

    case 0:
      break;
```

```
C:\Users\George katsaros\Documents\...\new era\...

Menu Heap:
==============
1-Eisagwgi
2-Diagrafi
3-Ektypwsi
4-HeapSort!
5-Eksodos
Epilogi?
5

--------------------------------
Process exited after 1.493 seconds with return value 5
Press any key to continue . . .
```

ΤΕΛΟΣ ΑΝΑΦΟΡΑΣ