

Practical Exercise 6

Ομάδα : 3

Συμμετέχοντες : Μιχάλης Μιχαήλ
Σώτος Βασιλείου
Πασιουρτίδης Κώστας



Άσκηση 1

```
int main()
{
    FILE *src = fopen("ECE325_numbers.txt", "r");
    int key, array[Nums], i=0, max, low, j, counter=0;
    while (!feof(src)) {
        fscanf(src, "%d ", &array[i]);
        // printf("%d ", array[i]);
        if (i < 1) {
            max=array[i];
            low=array[i];
        }

        if (array[i] > max)
            max=array[i];
        if (array[i] < low)
            low=array[i];

        i++;
    }

    fclose(src);
}
```

Αρχικά διαβάζω τους αριθμούς από το αρχείο και βρίσκω τον μικρότερο και μεγαλύτερο από τους αριθμούς όπου διαβάζω, επιπρόσθετα μετρώ τους αριθμούς που διάβασα με την μεταβλητή i.

```

int partition(int array[], int low, int high , int &counter) {

    // select the rightmost element as pivot
    int pivot = array[high];

    // pointer for greater element
    int i = (low - 1);

    // traverse each element of the array
    // compare them with the pivot
    for (int j = low; j < high; j++) {

        counter++; //2nd comparison here

        if (array[j] <= pivot) {
            // counter++; //3rd comparison here
            // if element smaller than pivot is found
            // swap it with the greater element pointed by i
            i++;

            // swap element at i with element at j
            swap(&array[i], &array[j]);
        }

        // swap the pivot element with the greater element at i
        swap(&array[i + 1], &array[high]);

        // return the partition point
        return (i + 1);
    }

}

void quickSort(int array[], int low, int high, int &counter) {
    if (low < high) {
        // counter++; //1st comparison here

        // find the pivot element such that
        // elements smaller than pivot are on left of pivot

```

Στον αλγόριθμο quicksort έχω τοποθετήσει 3 counter ωστόσο έχω ενεργοποίηση μόνο το 1 στο for του partition καθώς αυτές είναι οι συγκρίσεις για την μετακίνηση του αριθμού, τα άλλα 2 counter μετράνε τα swap όπου θα γίνουν εάν συμπεριλαμβάνονται και αυτά στις συγκρίσεις όπου μας ζητάτε μπορούν να ενεργοποιηθούν.

```

quickSort(array, low, max, counter);

char *filename = "sortednums.txt";
FILE *fp = fopen(filename, "a+");
for(j=0; j<i; j++){
    fprintf(fp, "%d\n", array[j]);
}

printf("File contain %d numbers\n", i );
printf("Quick sort comparisons %d", counter );

return 0;
}

```

Στην συνέχεια στέλνω στον αλγόριθμό quick sort τον πίνακα με τους αριθμούς και μου κάνει την ταξινόμηση και μου επιστρέφει τον αριθμό συγκρίσεων.

Τέλος δημιουργώ το file με τους ταξινομημένους αριθμούς και τυπώνω τον αριθμό των αριθμών και των συγκρίσεων.

Καλύτερη περίπτωση είναι $n \log n$ και χειρότερη n^2 .

Προγραμμα 2

Αρχικά στο κωδικα δηλώνω το file μου ECE325_dictionary.txt οπου θα παρω όλες τις λέξεις από το αρχείο. Δηλώνω πίνακα 10000 γραμμών ώστε να αποθηκεύσω όλα τα στοιχεία του text file και κατοπιν while loop που διαβάζουν μεχρι το τέλος του αρχείου γραμμη γραμμη αποθηκευονται ολες οι λέξεις του αρχείου.

Η μεταβλητη pl χρησιμοποιηται μετεπειτα όταν τερματιστει το προγραμμα στο τυπωμα του πληθους όλων των στοιχειων του text file.

Μεταβλητη pos για την θέση του στοιχειου που θα δώσει ο χρηστης αν υπαρχει.

User= με την λέξη που θα δώσει ο χρηστης

Στη συνεχεια εχω ένα flag and string ex=exit ώστε να με βοηθησει να τερματισω τον αλγοριθμο μου όταν ο χρηστης εισαγει exit.

Ο αλγοριθμος binarySearch είναι τυπου divide-and-conquer και βρισκει εάν η λέξη που εδωσε ο χρηστης είναι ιση με μια από τις λέξεις στο text file. Αν ισουνται τοτε επιστρεφει την θέση του στο αρχείο.

```
do{

    cout<<endl

    pos=binarySearch(words,0, pl-1, user)

    if (pos == -1)

        cout<<"Element is not present"<<endl

    else

        cout<<"Element present at position: "<<pos+1<<endl

        cout<<"dwse string"<<endl

        cin>>user

        if (user==ex)

            flag=false

    }while(flag!=false)
```

Αν η λέξη που εδωσε ο χρηστης υπαρχει στο αρχείο τοτε βρισκω την θέση του και τυπωνω αναλογα.

Στο τέλος τυπωνω και το πληθος του αρχείου ολο (το ζητα η ασκηση και είναι 9987).

Άσκηση 3

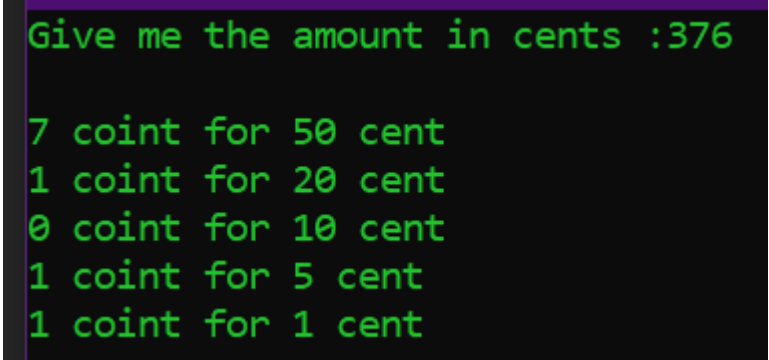
Sum = το ποσό που μένει να επιστραφεί
I=0;
int v[5] = {50,20,10,5,1}; //τι είδος
κέρματα έχω

```
While (sum != 0 ) {  
    While(sum>=v[i]){  
        Sum=sum - v[i];  
        result[i]++;  
    }  
    I++;  
}
```

Return result ;

Άπληστος αλγόριθμος πάντα επιλέγει αυτό που με βάση την τρέχουσα κατάσταση ,
δείχνει καλύτερο . => βγάζει πρώτα τα 50 cent

<https://medium.com/@gururajgpt544/greedy-algorithm-to-find-minimum-number-of-coins-6d364b40afd7>



```
Give me the amount in cents :376  
  
7 coin for 50 cent  
1 coin for 20 cent  
0 coin for 10 cent  
1 coin for 5 cent  
1 coin for 1 cent
```

1. For sorting n coins $O(n \log n)$.
2. While loop, the worst case is $O(\text{total})$. If all we have is the coin with 1-denomination.
3. Complexity for coin change problem becomes $O(n \log n) + O(\text{total})$.

while 72-22

após
resposta
=

while

while

1. $\frac{50}{x}$

2. $x \mid \frac{20}{y} \rightarrow \text{per } 20 \text{ per}$

3. $y \mid \frac{10}{z} \rightarrow \text{per } 10 \text{ per}$

4. $z \mid \frac{5}{f} \rightarrow \text{per } 5 \text{ per}$

5. $f \mid \frac{1}{f} \rightarrow \text{per } 1 \text{ per}$