ALGORİTMA ANALİZİ

ÖDEV 1 | SORU 1-A

GRUP 1

İrem ATILGAN

17061036

[illegible]

07.11.2020

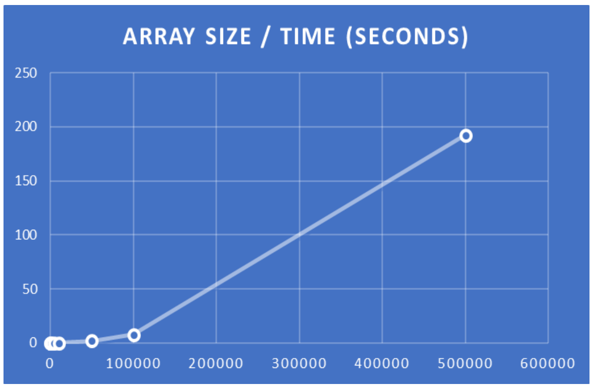
SORU 1

N elemanlı bir dizide birbirine en yakın değere sahip iki elemanın bulunması isteniyor.

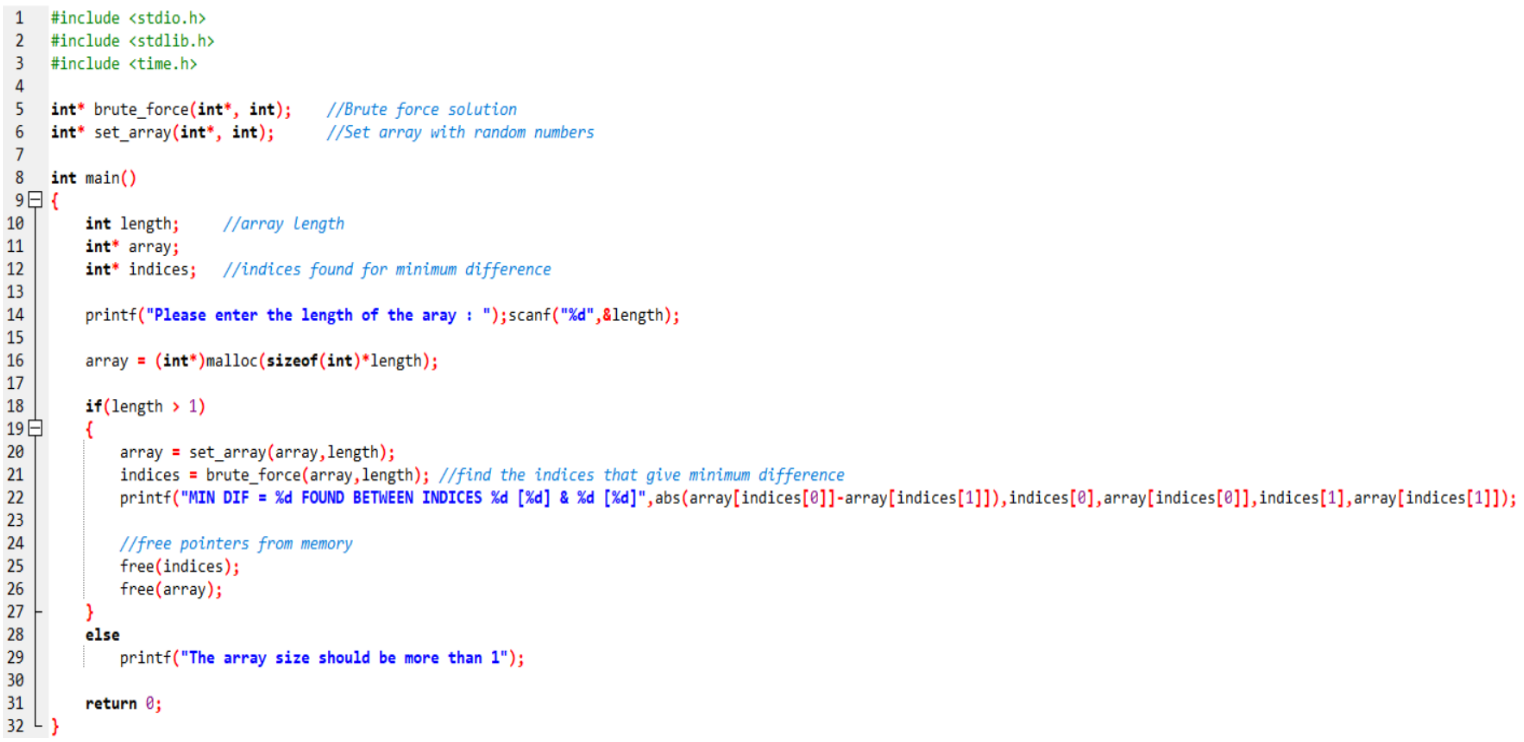
A) Brute Force Yaklaşımı : Brute Force Yaklaşımı'nda dizide bulunan her bir elemanın, dizideki diğer elemanlarla farkı bulunmuş ve minimum fark ile karşılaştırılmıştır. Dizide N eleman olduğundan ve her eleman için N elemanla olan farkına (elemanın kendi kendisinin farkı alınmasa da tüm hücreler geziliyor) bakıldığından karmaşıklığı :

Best Case : $O(N^2)$
Average Case : $\theta(N^2)$
Worst Case : $\Omega(N^2)$

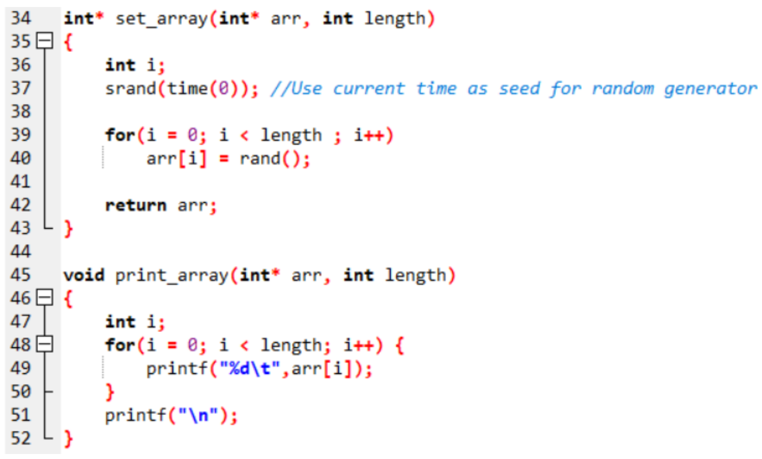
Array Size	Time (Seconds)
10	0
100	0
1000	0.001
5000	0.027
10000	0.085
50000	1.982
100000	7.858
500000	192.19



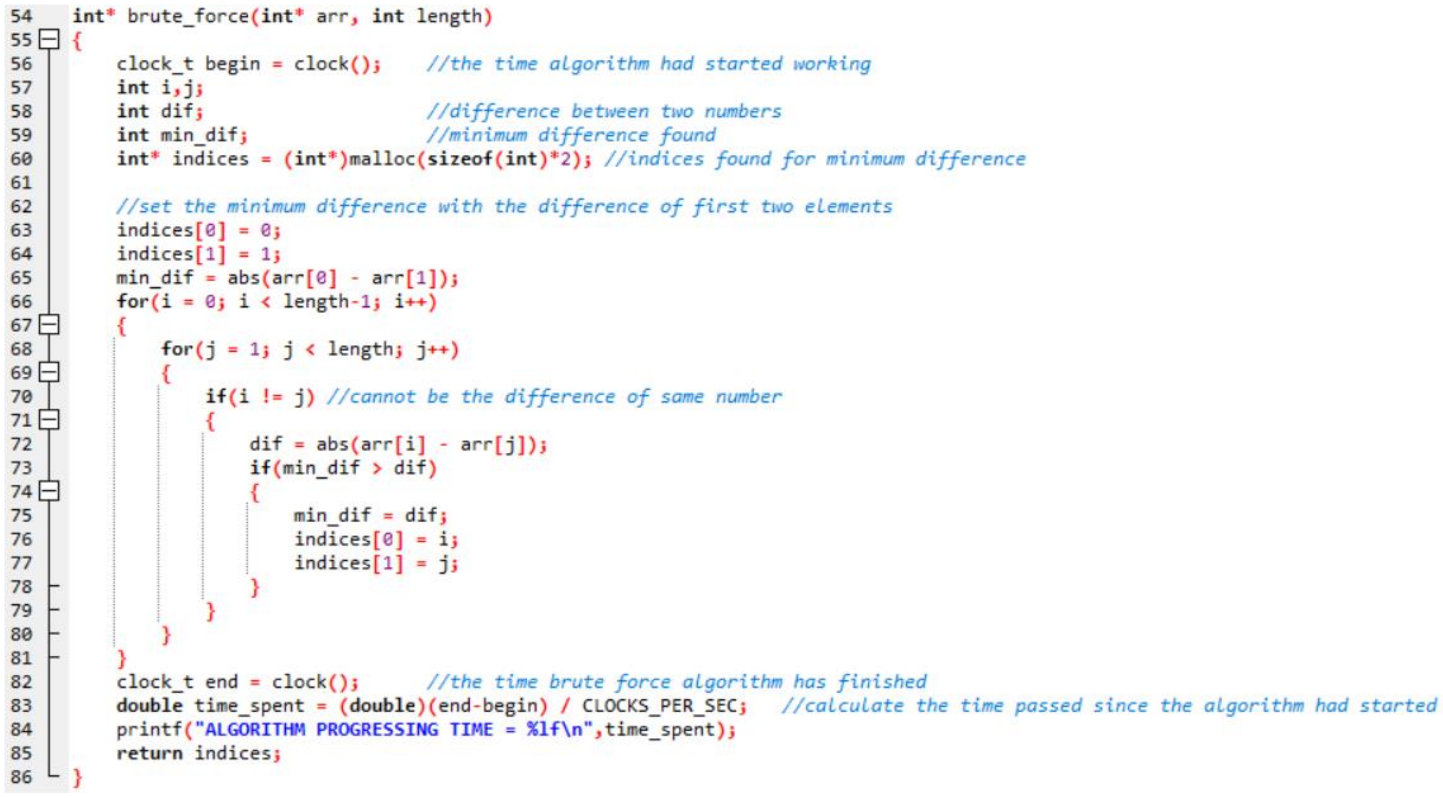
PROGRAM KODLARI



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

int* brute_force(int*, int);    //Brute force solution
int* set_array(int*, int);      //Set array with random numbers

int main()
{
    int length;     //array length
    int* array;
    int* indices;   //indices found for minimum difference

    printf("Please enter the length of the aray : ");scanf("%d",&length);

    array = (int*)malloc(sizeof(int)*length);

    if(length > 1)
    {
        array = set_array(array,length);
        indices = brute_force(array,length); //find the indices that give minimum difference
        printf("MIN DIF = %d FOUND BETWEEN INDICES %d [%d] & %d [%d]",abs(array[indices[0]]-array[indices[1]]),indices[0],array[indices[0]],indices[1],array[indices[1]]);

        //free pointers from memory
        free(indices);
        free(array);
    }
    else
        printf("The array size should be more than 1");

    return 0;
}
```

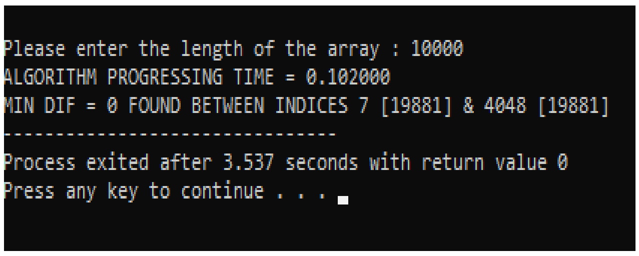


```c
int* set_array(int* arr, int length)
{
    int i;
    srand(time(0)); //Use current time as seed for random generator

    for(i = 0; i < length ; i++)
        arr[i] = rand();

    return arr;
}

void print_array(int* arr, int length)
{
    int i;
    for(i = 0; i < length; i++) {
        printf("%d\t",arr[i]);
    }
    printf("\n");
}
```

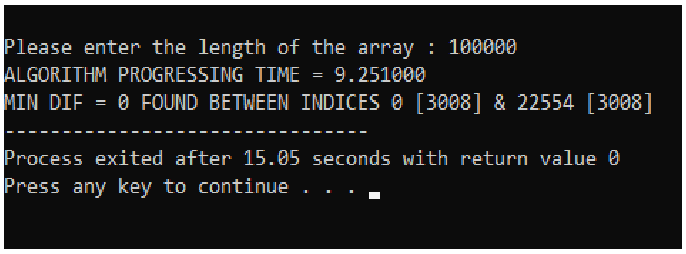


```c
int* brute_force(int* arr, int length)
{
    clock_t begin = clock();     //the time algorithm had started working
    int i,j;
    int dif;                     //difference between two numbers
    int min_dif;                 //minimum difference found
    int* indices = (int*)malloc(sizeof(int)*2); //indices found for minimum difference

    //set the minimum difference with the difference of first two elements
    indices[0] = 0;
    indices[1] = 1;
    min_dif = abs(arr[0] - arr[1]);
    for(i = 0; i < length-1; i++)
    {
        for(j = 1; j < length; j++)
        {
            if(i != j) //cannot be the difference of same number
            {
                dif = abs(arr[i] - arr[j]);
                if(min_dif > dif)
                {
                    min_dif = dif;
                    indices[0] = i;
                    indices[1] = j;
                }
            }
        }
    }
    clock_t end = clock();      //the time brute force algorithm has finished
    double time_spent = (double)(end-begin) / CLOCKS_PER_SEC;    //calculate the time passed since the algorithm had started
    printf("ALGORITHM PROGRESSING TIME = %lf\n",time_spent);
    return indices;
}
```

EKRAN ÇIKTILARI



```
Please enter the length of the array : 10000
ALGORITHM PROGRESSING TIME = 0.102000
MIN DIF = 0 FOUND BETWEEN INDICES 7 [19881] & 4048 [19881]
--------------------------------
Process exited after 3.537 seconds with return value 0
Press any key to continue . . .
```



```
Please enter the length of the array : 100000
ALGORITHM PROGRESSING TIME = 9.251000
MIN DIF = 0 FOUND BETWEEN INDICES 0 [3008] & 22554 [3008]
--------------------------------
Process exited after 15.05 seconds with return value 0
Press any key to continue . . .
```