Ders 24

Örnek ödev essiz sayıyı bul

```
#define         SIZE        20

int main()
{
    int a[SIZE];

    randomize();

    for (int i = 0; i < SIZE; ++i) {
        a[i] = rand() % 20;
    }

    print_array(a, SIZE);

    int k;
    for (int i = 0; i < SIZE; ++i) {
        for (k = 0; k < SIZE; ++k) {
            if (i != k && a[i] == a[k]) {
                break;
            }
        }
        if (k == SIZE) {
            printf(_Format:"%3d " ,a[i]);
        }
```

Sayaçsızda döngü değişkeni bayrak olarak kullandık.

Diğer türlü çözmek

```c
#define         SIZE        20

int main()
{
    int a[SIZE];

    randomize();

    for (int i = 0; i < SIZE; ++i) {
        a[i] = rand() % 20;
    }

    print_array(a, SIZE);

    int cnt[20] = { 0 };

    for (int i = 0; i < SIZE; ++i) {
        ++cnt[a[i]];
    }

    for (int i = 0; i < SIZE; ++i) {
        if (cnt[i] == 1)
            printf("%3d ", i);
    }

    printf("\n");
```

1 dizi ekleyerek o(n) karmaşıklığında çözüldü.

Ödev sorusu

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

#define         SIZE      100

int main()
{
    int a[SIZE];

    randomize();

    for (int i = 0; i < SIZE; ++i) {
        a[i] = rand() % 3;
    }

    print_array(a, SIZE);
}
```

Dizi 0 ve 1 lerden oluşmakta bu diziyi o(n)

Karmaşıklığında yazınız .

Yöntem 0 1 2 sayılarının adeti kadarını tut ve sonra bi daha yazdır.

Yöntem 2 farklı çöz

```c
#define     URAND_MAX               20

//her çağrıldığında 0 dahil URAND_MAX [0 URAND_MAX)
// eger cagrilabileceğinden daha fazla cagrilirsa işlev hata kod

int urand(void)
{
    static int flags[URAND_MAX] = { 0 };
    static int cnt = 0;

    if (cnt == URAND_MAX) {
        return -1;
    }

    int val;

    while (1) {
        val = rand() % URAND_MAX;
        if (flags[val] == 0)
            break;
    }

    ++cnt;
    flags[val] = 1;

    return val;
}


int main()
{
    randomize();

    for (int i = 0; i < URAND_MAX; ++i) {
```

Örnek dizideki sayılar birbirinden farklı ve rasgele olsun

```c
#define     URAND_MAX               20

//her çağrıldığında 0 dahil URAND_MAX [0 URAND_MAX)
// eger cagrilabileceğinden daha fazla cagrilirsa işlev hata kod

int urand(void)
{
    static int flags[URAND_MAX] = { 0 };
    static int cnt = 0;

    if (cnt == URAND_MAX) {
        return -1;
    }

    int val;

    while (1) {
        val = rand() % URAND_MAX;
        if (flags[val] == 0)
            break;
    }

    ++cnt;
    flags[val] = 1;

    return val;
}


int main()
{
    randomize();

    for (int i = 0; i < URAND_MAX; ++i) {
```

Bir idiom aynı işi yapan while

```c
int val;

while (flags[val = rand() % URAND_MAX])
    ;

++cnt;
flags[val] = 1;

return val;
```

Sırlama algoritmaları

Örnek algoritma bubble sort

```
bubble sort

insertion sort

856 678 892 413 926 980 472 104 355 131
678 856 413 892 926 980 472 104 355 131
678 856 413 892 926 472 980 104 355 131
678 856 413 892 926 472 104 982 355 131
678 856 413 892 926 472 104 355 982 131

678 856 413 892 926 472 104 355 131 980
                                926
```

```
#define         SIZE        100

int main()
{
    int a[SIZE];

    randomize();
    set_array_random(a, SIZE);
    print_array(a, SIZE);

    for (int i = 0; i < SIZE - 1; ++i) {
        for (int k = 0; k < SIZE - 1; ++k) {
            if (a[k] > a[k
                int temp =
                a[k] = a[k + 1];
                a[k + 1] = temp;
            }
        }
    }

    print_array(a, SIZE);

}
```

(local variable) int k
Search Online

İç deki döngü her tur fazladan bir basamak dönmesin diye 2. Forun içinde i kadar çıkartıyırız.

```c
int main()
{
    int a[SIZE];

    randomize();
    set_array_random(a, SIZE);
    print_array(a, SIZE);

    for (int i = 0; i < SIZE - 1; ++i) {
        for (int k = 0; k < SIZE - 1 - i; ++k) {
            if (a[k] > a[k + 1]) {
                int temp = a[k];
                a[k] = a[k + 1];
                a[k + 1] = temp;
            }
        }
    }

    print_array(a, SIZE);
```

Büyükten küçüğe sıralamak içinde if içini < > değiştirmek yeterli

Tek olanlar bir yerde çiftler bir yerde

```c
#define  SIZE     10
```

```c
int main() {

    int a[SIZE];

    randomize();

    set_random_array(a,SIZE);
    print_array(a, SIZE);

        for (int i = 0; i < SIZE - 1; ++i){
            for (int k = 0; k < SIZE - 1 - i; ++k) {

                if ( (a[k]%2 == 0 && a[k+1] %2) || (a[k] % 2 == a[k + 1] % 2)&& a[k] > a[k+1]) {

                    int temp = a[k];
                    a[k] = a[k + 1];
                    a[k + 1] = temp;

                }
            }
```

İf içini değiştirdik ve sıralama değişti bu bubble sortalgoritması. İstek değişti algoritma aynı.

Örnek

o(n^2) ve o(nlogn) kullanarak dizi sıralama karşılaştırma yap.

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

#define         SIZE        3'000'000

int main()
{
    int* pd = (int*)malloc(_Size:SIZE * sizeof(int));
    if (!pd) {
        printf(_Format:"bellek yetersiz\n");
        return 1;
    }

    set_array_random(pd, SIZE);

    clock_t start = clock();

    for (int i = 0; i < SIZE - 1; ++i) {
        for (int k = 0; k < SIZE - 1 - i; ++k) {
            if (pd[k] > pd[k + 1]) {
                int temp = pd[k];
                pd[k] = pd[k + 1];
                pd[k + 1] = temp;
            }
        }
    }

    clock_t end = clock();

    printf(_Format:"siralama tamamlandi sure %.6f saniye\n", (double)(end - start) / CLOCKS_PER_SEC);
    (void)getchar();
    print_array(pd, SIZE);
    free(_Block:pd);

}
```

1000000 u yarım saatte yapıyor.

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

#define         SIZE        10'000'000

int icmp(const void* vp1, const void* vp2)
{
    return *(const int*)vp1 - *(const int*)vp2;
}


int main()
{
    int* pd = (int*)malloc(SIZE * sizeof(int));
    if (!pd) {
        printf("bellek yetersiz\n");
        return 1;
    }

    set_array_random(pd, SIZE);

    printf("siralama basladi\n");
    clock_t start = clock();
    qsort(pd, SIZE, sizeof(int), &icmp);

    /*for (int i = 0; i < SIZE - 1; ++i) {
        for (int k = 0; k < SIZE - 1 - i; ++k) {
            if (pd[k] > pd[k + 1]) {
                int temp = pd[k];
                pd[k] = pd[k + 1];
                pd[k + 1] = temp;
            }
        }
    }*/

    clock_t end = clock();

    printf("siralama tamamlandi sure %.6f saniye\n", (double)(end - start) / CLOCKS_PER_SEC);
    (void)getchar();
```

Bu ise aynı işlemi 0.1 saniyede yapmakta karmaşıklık azalınca süre çok fazla azaldı.

Binary sort

```
idx_first      idx_last
idx_mid = (idx_first + idx_last) / 2

if (a[idx_mid] == key)

if (a[idx_mid] > key)
    idx_last = idx_mid - 1;
else
    idx_first = idx_mid + 1;
```

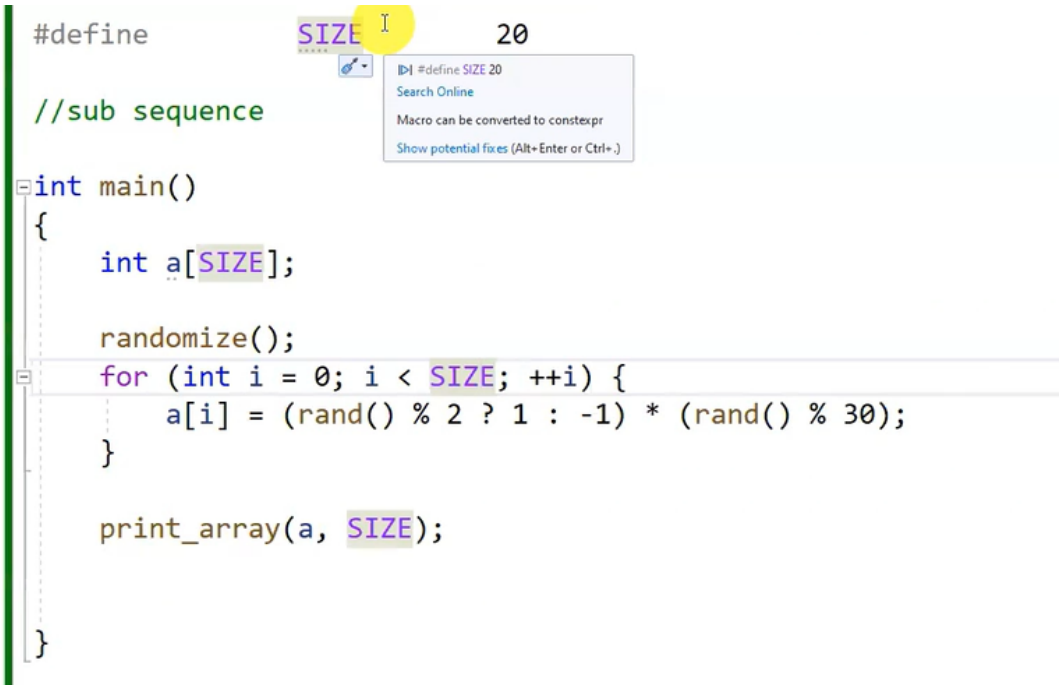
```
#define _CRT_SECURE_NO_WARNINGS

#include "nutility.h"
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

#define         SIZE        10000
```

```
#define         SIZE        10000


int main()
{
    int a[SIZE];

    randomize();
    set_array_random(a, SIZE);
    sort_array(a, SIZE);
    //print_array(a, SIZE);

    int key;
    printf("aranacak degeri giriniz: ");
    scanf("%d", &key);
    int idx_first = 0;
    int idx_last = SIZE - 1;
    int idx_mid;
    int cnt = 0;
    while (idx_first <= idx_last) {
        ++cnt;
        idx_mid = (idx_first + idx_last) / 2;
        if (a[idx_mid] == key)
            break;

        if (a[idx_mid] > key) {
            idx_last = idx_mid - 1;
        }
        else {
            idx_first = idx_mid + 1;
        }
    }

    if (idx_first > idx_last) {
        printf("bulunamadi cnt = %d\n", cnt);
    }
    else {
        printf("bulundu dizinin %d. indisli elemani\n", idx_mid);
    }


}
```

Ödev sorusu zor soru sub sequence

MAXİMUM sub-sequence problem

28 -26 -4 4 -2 -21 -27 -28 17 0 25 9 9 9 0 -15 -28 9 -18 -14



Ödevi çözünüz defterde açıklaması vardır.

Merge algoritması

Ben kendim yapmıştım

Örnek merge kodu

```c
#include<stdio.h>

int main() {
    int counta =0;
    int countb = 0;
    int a[7] = { 2,7,9,23,45,78,0 };
    int b[7] = { 3,6,45,67,80,90,0 };
    int c[12] = {0};
    int n=12;
    while (n--) {
        if (counta > 5) {

            c[counta + countb] = b[countb];
            countb++;

        }
        else if (countb > 5) {

            c[counta + countb] = a[counta];
            counta++;

        }

        else if ((a[counta] <= b[countb]))
        {
            c[counta + countb] = a[counta];
                counta++;
        }
        else if(a[counta] > b[countb])
        {
            c[counta + countb] = b[countb];
            countb++;
        }

    }
    else if(a[counta] > b[countb])
    {
        c[counta + countb] = b[countb];
        countb++;
    }


}
for (int i = 0; i < 12; i++)
    printf(" %d ,", c[i]);
```

HOCANIN YAPTIĞI ÖRNEK

Algoritmamız aynı ama o forile yaptı indisi i ile yazdırdı count artışını dizinin adının içinde yaptı.

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


#define         SIZE        10

//sub sequence

int main()
{
    int a[SIZE];
    int b[SIZE];
    int c[2 * SIZE];

    randomize();
    set_array_random(a, SIZE);
    sort_array(a, SIZE);
    set_array_random(b, SIZE);
    sort_array(b, SIZE);
    print_array(a, SIZE);
    print_array(b, SIZE);

    int idx_a = 0;
    int idx_b = 0;

    for (int i = 0; i < 2 * SIZE; ++i) {
        if (idx_a == SIZE) {
            c[i] = b[idx_b++];
        }
        else if (idx_b == SIZE) {
            c[i] = a[idx_a++];
        }
        else if (a[idx_a] < b[idx_b]) {
            c[i] = a[idx_a++];
        }
        else {
            c[i] = b[idx_b++];
        }
    }

    print_array(c, 2 * SIZE);
}
```

Partition algoritması örnek ödev !!!!!!!!!!!!!!