

# HW10

## *Dynamic Array Adımları :*

- 1-)Data txt dosyasını okuma modunda açtım.
- 2-)Arrayime kendi belirlediğim hafıza boyutuna göre malloc ile bellekte yer ayırttım.
- 3-)Text dosyasından bir tane long int alınır ve while döngüsüne girilir.While döngüsü parametre olarak girilen finalNum sayısı kadar döner.Mesela finalNum 500000 ise data.txt dosyasından 500000 sayısına kadar long int alınır.
- 4-)Her alınan sayının asal olup olmadığı kontrol edilir.[Asal Alma Fonksiyonum : eğer bir sayı asalsa 2 den onun karaköküne kadar olan sayılardan en az birine bölünür.Asal olan sayılar ise bu aralıktaki hiç bir sayıya bölünmez.
- 5-)Used adlı bir değişken belirledim bu değişken hem arrayin indexini hemde bellekte kullanılan yer sayısını tutacak.Used değişkeninin ilk değeri sıfırdır ve her asal olan sayıyı bulunduğca bir artırılır.
- 6-)Eğer kullanılan arrayin (yani usedin) bir fazlası array için bellekten ayırtılmış yere denk geliyorsa bellek boyutu iki katına çıkarılır ve ek olarak tanımladığım array için bu bellek boyutu kadar yer alınır.Daha sonra ana arrayimin elemanlarını tutması için elemanları ikinci arraye atanır.Bellekte yer kaplamaması için ana arrayim için aldığım ilk yeri free ile belleğe geri veririm.Son olarak da ana arrayimin yerini ikinci olarak eklediğim arrayin yeri olarak gösteririm.
- 7-)Dosyanın sonundaki EOF un alınmaması için eğer dosyadan alınan bir satırda 3 tane eleman yoksa (yani %ld %c %d) number sayısı 10 artırılarak finalNum dan daha büyük olur ve döngüden çıkar.
- 8-)İşlemler bittiğinde data.txt dosyası kapatılır.
- 9-)Eğer parametre olarak girilen digit sayısı 1 ise dosyadaki asal sayılar parametre olarak gönderilen outputFile ın gösterdiği dosyaya yazdırılır.[Digit Num : parametre olarak gönderilen 1 ve ya 0 değerleridir.Bu değerler asal sayıların yazdırılıp yazdırılmayacağını belirler.1 sayısı yazdırılacağını,0 ise bastırılmayacağını söyler.
- 10-)Fonksiyonun sonunda yer ayırtılan array belleğe geri verilir.

### *Linked List Adımları :*

- 1-)Data txt dosyası okuma modunda açılır.
- 2-)Text dosyasından bir tane long int alınır ve while döngüsüne girilir.While döngüsü parametre olarak girilen finalNum sayısı kadar döner.Mesela finalNum 500000 ise data.txt dosyasından 500000 sayısına kadar long int alınır.
- 3-)Her alınan sayının asal olup olmadığı kontrol edilir.
- 4-)Yeni eklenecek olan node un datası dosyadan yeni alınan asal sayıdır ve next ide son node olduğu için NULL olur.
- 5-)Eğer başta linked list boş ise dosyadan alınan sayı ile ilk node oluşturulur.
- 6-)Eğer eklenecek sayının önünde başka node lar var ise son node u ifade eden lastNode değişkeni artık yeni eklenecek olan sayının node unu gösterir.
- 7-)Dosyanın sonundaki EOF un alınmaması için eğer dosyadan alınan bir satırda 3 tane eleman yoksa (yani %ld %c %d) number sayısı 10 artırılarak finalNum dan daha büyük olur ve döngüden çıkar.
- 8-)Digit Number 1 ise linked listin elemanları bastırılır,0 ise bastırılmaz.
- 9-)Fonksiyonun sonunda ise node lar free ile belleğe geri verilir.

### *Main Adımları :*

- 1-)Clock fonksiyonu ile zaman hesaplanır : fonksiyona gitmeden ve gittikten sonraki zamanlar birbirinden çıkarılır ve time.h kütüphanesinin sabiti olan CLOCKS\_PER\_SEC a bölünür.clock fonksiyonu zamanı saniye cinsinden return ettiği için çıkan zaman sonucu 1000 ile çarpılarak milisaniye cinsine dönüştürülür.
- 2-)Fonksiyonların çalıştıkları zamanlar output\_prime\_dynamic\_array.txt ve output\_prime\_LiknedList.txt dosyalarına yazılır.Ayreten bu sonuçlar mainde de basılır.
- 3-)Yazdırılma işlemleri bittikten sonra dosyalar kapanır.

## DYNAMIC ARRAY DOSYA ÇIKTISI(output\_prime\_dynamic\_array.txt)

```
output_prime_dynamic_array.txt (-/Desktop) - gedit
Open  x  hw10.c  x  output_prime_LiknedList.txt  x  output_prime_dynamic_array.txt  x  zeynep_yukse1_161044068-1.c  x  Save

78456 999331,00
78457 999359,00
78458 999371,00
78459 999377,00
78460 999389,00
78461 999431,00
78462 999433,00
78463 999437,00
78464 999451,00
78465 999491,00
78466 999499,00
78467 999521,00
78468 999529,00
78469 999541,00
78470 999553,00
78471 999563,00
78472 999599,00
78473 999611,00
78474 999613,00
78475 999623,00
78476 999631,00
78477 999653,00
78478 999667,00
78479 999671,00
78480 999683,00
78481 999721,00
78482 999727,00
78483 999749,00
78484 999763,00
78485 999769,00
78486 999773,00
78487 999809,00
78488 999853,00
78489 999863,00
78490 999883,00
78491 999907,00
78492 999917,00
78493 999931,00
78494 999953,00
78495 999959,00
78496 999961,00
78497 999979,00
78498 999983,00
78499
78500 *****TIME CALCULATOR RESULTS*****
78501
78502 Time for 1.000.000(with writing prime number into file) ==> 1191.315000 milliseconds
78503 Time for 1.000.000 ==> 1112.020000 milliseconds
78504 Time for 750.000 ==> 778.175000 milliseconds
78505 Time for 500.000 ==> 466.601000 milliseconds

Plain Text  Tab Width: 4  Ln 78457, Col 10  INS
```

## LINKED LIST DOSYA ÇIKTISI(output\_prime\_LinkedList.txt)

```
output_prime_LinkedList.txt (-/Desktop) - gedit
Open  x  hw10.c  x  output_prime_LinkedList.txt  x  output_prime_dynamic_array.txt  x  Save

78456 999331,00
78457 999359,00
78458 999371,00
78459 999377,00
78460 999389,00
78461 999431,00
78462 999433,00
78463 999437,00
78464 999451,00
78465 999491,00
78466 999499,00
78467 999521,00
78468 999529,00
78469 999541,00
78470 999553,00
78471 999563,00
78472 999599,00
78473 999611,00
78474 999613,00
78475 999623,00
78476 999631,00
78477 999653,00
78478 999667,00
78479 999671,00
78480 999683,00
78481 999721,00
78482 999727,00
78483 999749,00
78484 999763,00
78485 999769,00
78486 999773,00
78487 999809,00
78488 999853,00
78489 999863,00
78490 999883,00
78491 999907,00
78492 999917,00
78493 999931,00
78494 999953,00
78495 999959,00
78496 999961,00
78497 999979,00
78498 999983,00
78499
78500 *****TIME CALCULATOR RESULTS LINKED LIST*****
78501
78502 Time for 1.000.000(with writing prime number into file) ==> 1137.007000 milliseconds
78503 Time for 1.000.000 ==> 1126.453000 milliseconds
78504 Time for 750.000 ==> 761.009000 milliseconds
78505 Time for 500.000 ==> 475.348000 milliseconds

Plain Text  Tab Width: 4  Ln 76688, Col 10  OVR
```

# TERMİNAL ÇIKTISI

```
osboxes@osboxes: ~/Desktop
Time for 500.000 ==> 444.083000 milliseconds
osboxes@osboxes:~/Desktop$ ./exe
*****TIME CALCULATOR RESULTS FOR DYNAMIC ARRAY*****
Time for 1.000.000(with writing prime number into file) ==> 1145.852000 milliseconds
Time for 1.000.000 ==> 1169.547000 milliseconds
Time for 750.000 ==> 788.860000 milliseconds
Time for 500.000 ==> 469.995000 milliseconds
*****TIME CALCULATOR RESULTS FOR LINKED LIST*****
Time for 1.000.000(with writing prime number into file) ==> 1154.050000 milliseconds
Time for 1.000.000 ==> 1147.699000 milliseconds
Time for 750.000 ==> 783.613000 milliseconds
Time for 500.000 ==> 461.788000 milliseconds
osboxes@osboxes:~/Desktop$ ./exe
*****TIME CALCULATOR RESULTS FOR DYNAMIC ARRAY*****
Time for 1.000.000(with writing prime number into file) ==> 1151.417000 milliseconds
Time for 1.000.000 ==> 1147.699000 milliseconds
Time for 750.000 ==> 763.273000 milliseconds
Time for 500.000 ==> 483.297000 milliseconds
*****TIME CALCULATOR RESULTS FOR LINKED LIST*****
Time for 1.000.000(with writing prime number into file) ==> 1148.346000 milliseconds
Time for 1.000.000 ==> 1125.460000 milliseconds
Time for 750.000 ==> 762.775000 milliseconds
Time for 500.000 ==> 470.202000 milliseconds
osboxes@osboxes:~/Desktop$ ./exe
*****TIME CALCULATOR RESULTS FOR DYNAMIC ARRAY*****
Time for 1.000.000(with writing prime number into file) ==> 1135.203000 milliseconds
Time for 1.000.000 ==> 1126.885000 milliseconds
Time for 750.000 ==> 771.068000 milliseconds
Time for 500.000 ==> 454.623000 milliseconds
*****TIME CALCULATOR RESULTS FOR LINKED LIST*****
Time for 1.000.000(with writing prime number into file) ==> 1145.033000 milliseconds
Time for 1.000.000 ==> 1122.599000 milliseconds
Time for 750.000 ==> 784.314000 milliseconds
Time for 500.000 ==> 446.904000 milliseconds
osboxes@osboxes:~/Desktop$ ./exe
*****TIME CALCULATOR RESULTS FOR DYNAMIC ARRAY*****
Time for 1.000.000(with writing prime number into file) ==> 1166.282000 milliseconds
Time for 1.000.000 ==> 1099.918000 milliseconds
Time for 750.000 ==> 791.334000 milliseconds
Time for 500.000 ==> 478.179000 milliseconds
*****TIME CALCULATOR RESULTS FOR LINKED LIST*****
Time for 1.000.000(with writing prime number into file) ==> 1137.007000 milliseconds
Time for 1.000.000 ==> 1126.453000 milliseconds
Time for 750.000 ==> 761.809000 milliseconds
Time for 500.000 ==> 475.348000 milliseconds
osboxes@osboxes:~/Desktop$
```

NOT

1.000.000 ilk çalıştırıldığında asal sayıları dosyaya da yazdırılacağı için daha yavaştır.