

Bilişim Sistemler Mühendisliği

|  |  |
| --- | --- |
| Öğrenci No: | B191200370 |
| Öğrenci Ad Soyad: | Esra AKSU |
| Ders: | Veri Yapıları |
| Ödev: | Stackler |

Projenin Amacı

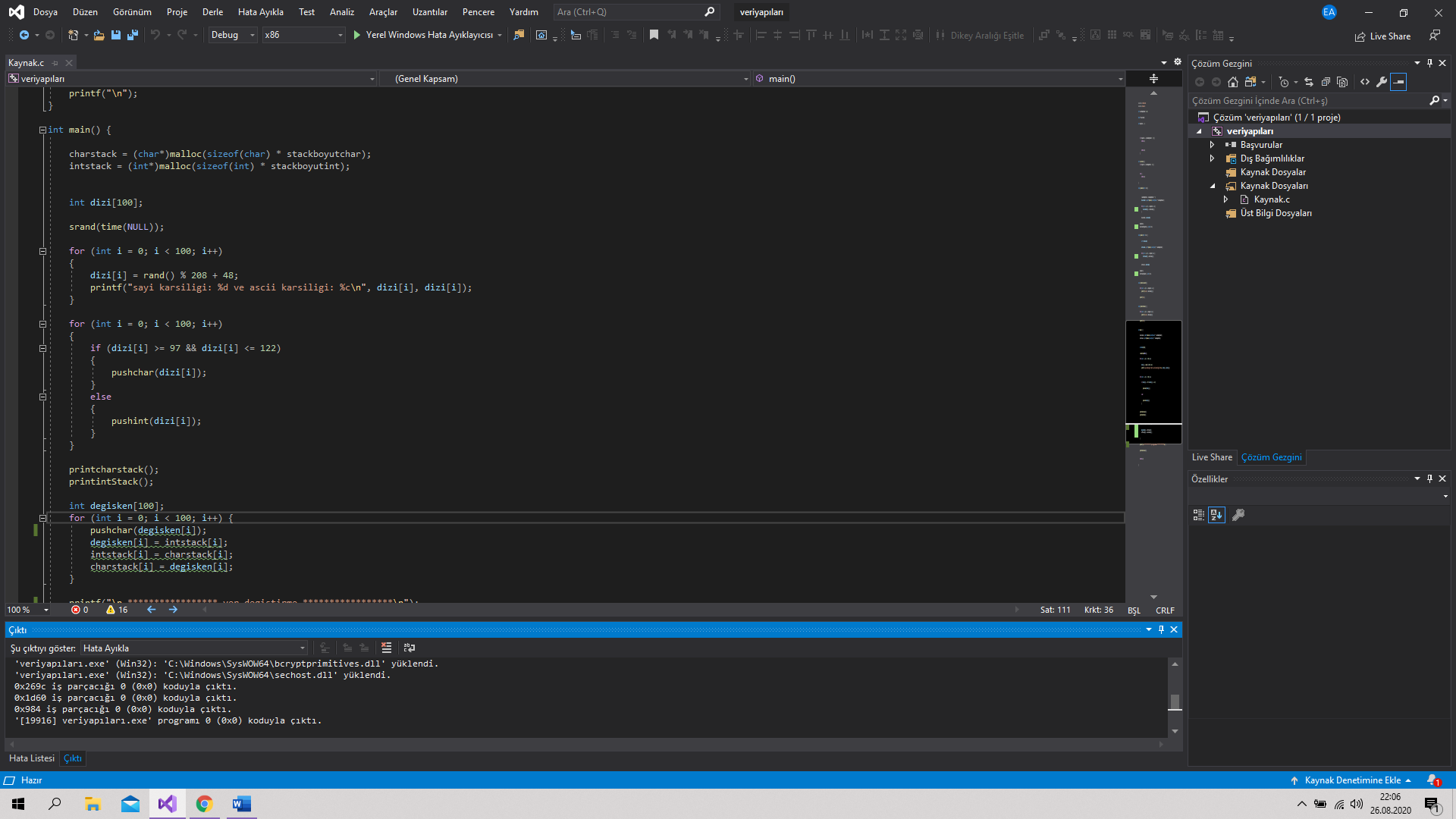
100 adet 48-255 arası rastgele sayı üretilmesi sağlanacaktır. Üretilen sayılara ait ASCII kod küçük harf karakterlerine karşılık geliyorsa char stack’ine gelmiyorsa int stack’ine atılmalıdır.

Başlangıçta stack’ler oluşturulurken stacklerin boyutu her öğrencinin numarasının son rakamı kadar (numarası 0 ile bitenler 10) olmalı, stack dolu ise ve yeni elaman eklenme ihtiyacı da varsa stack boyutu var olan boyutun iki katına çıkarılmalıdır.

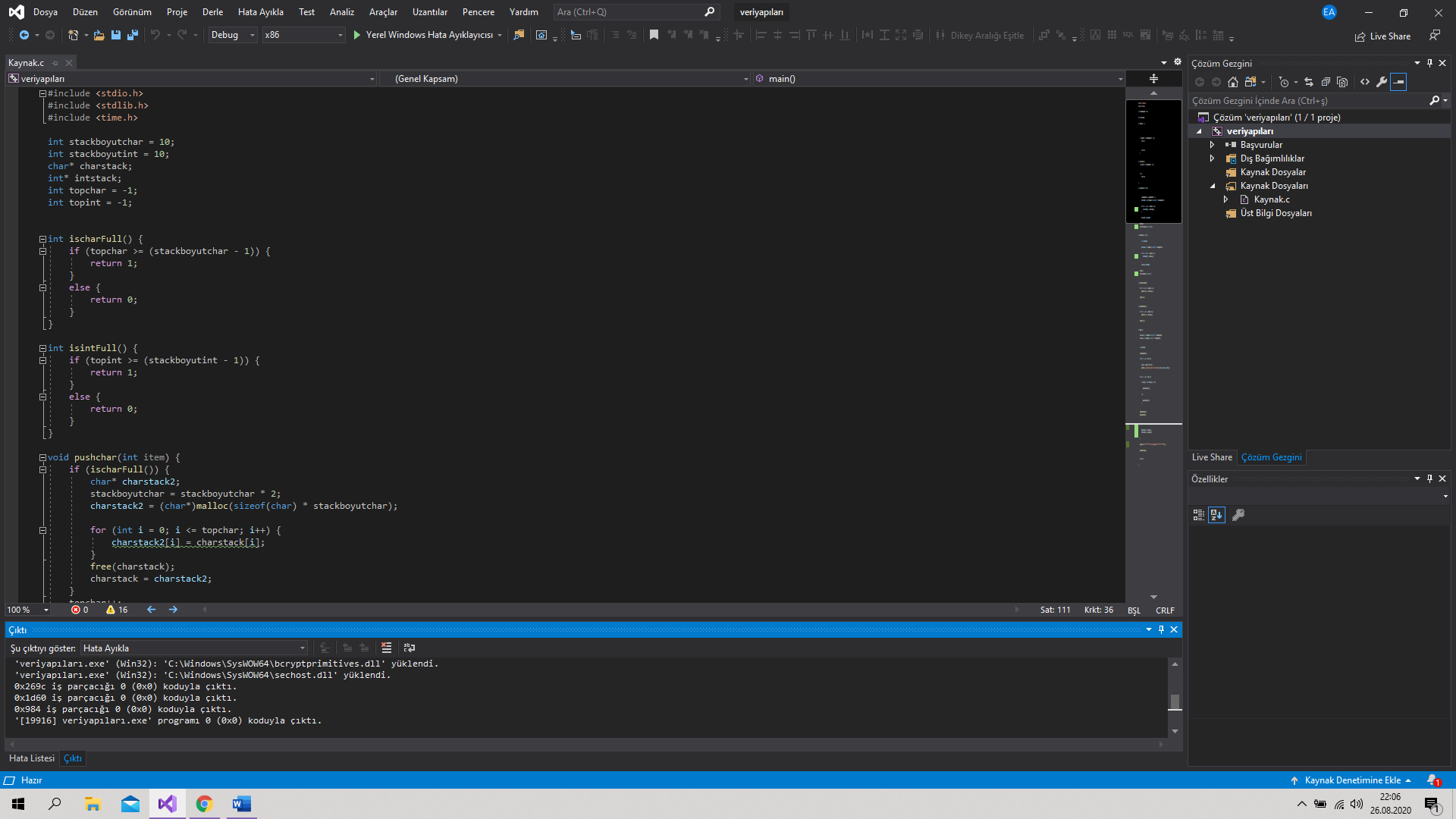
Tüm işlem tamamlandığında her iki stack’e ait verilerin yer değiştirilmesi istenmektedir. (yer değiştirmeden sonra tüm elemanların sırası korunmalıdır).

Projenin Yapım Aşaması

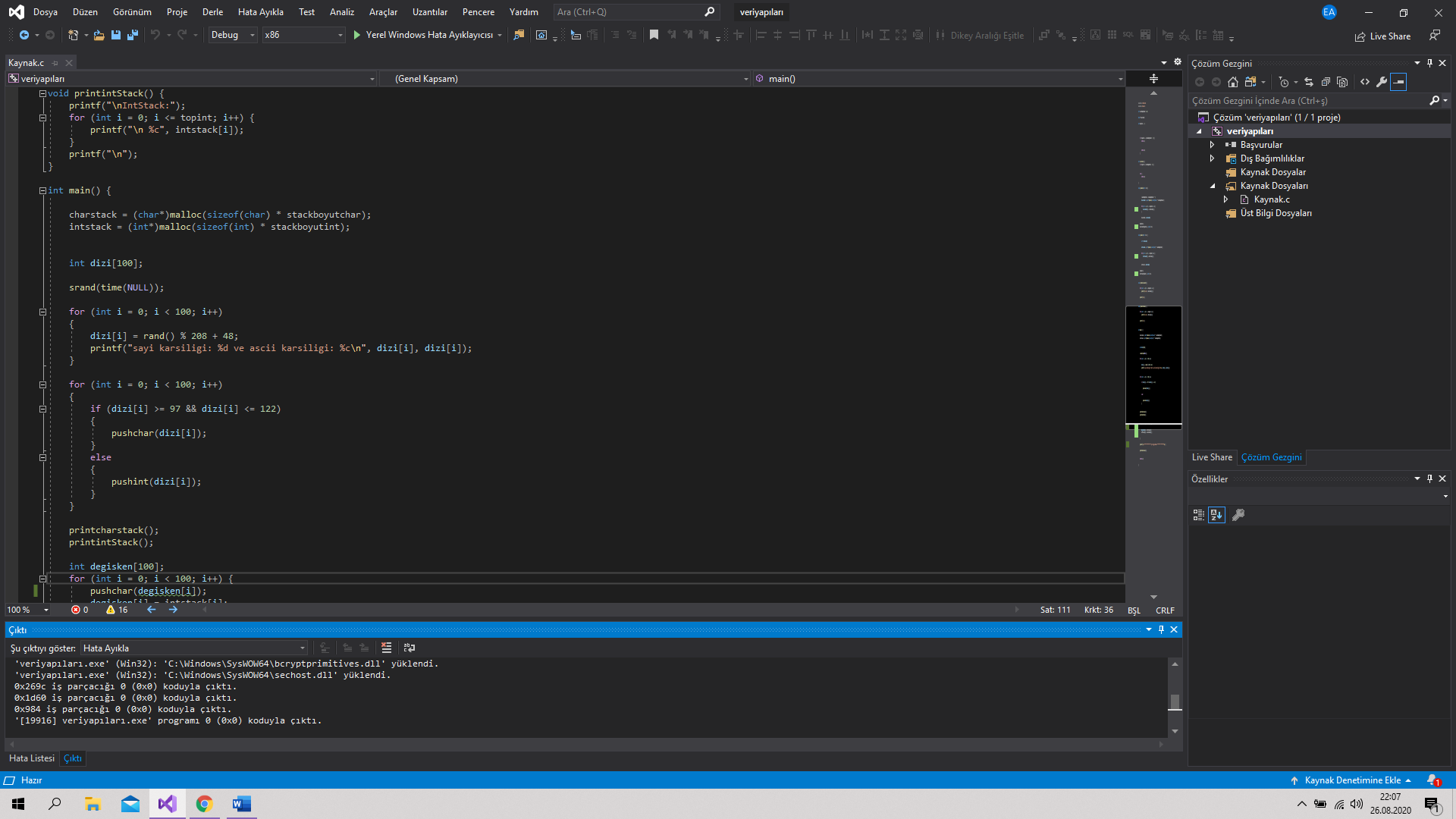
1. Adım: 48-255 aralığında rastgele atanacak şekilde 100 adet sayı üretilmesi için main içerisinde dizi tanımlanır.



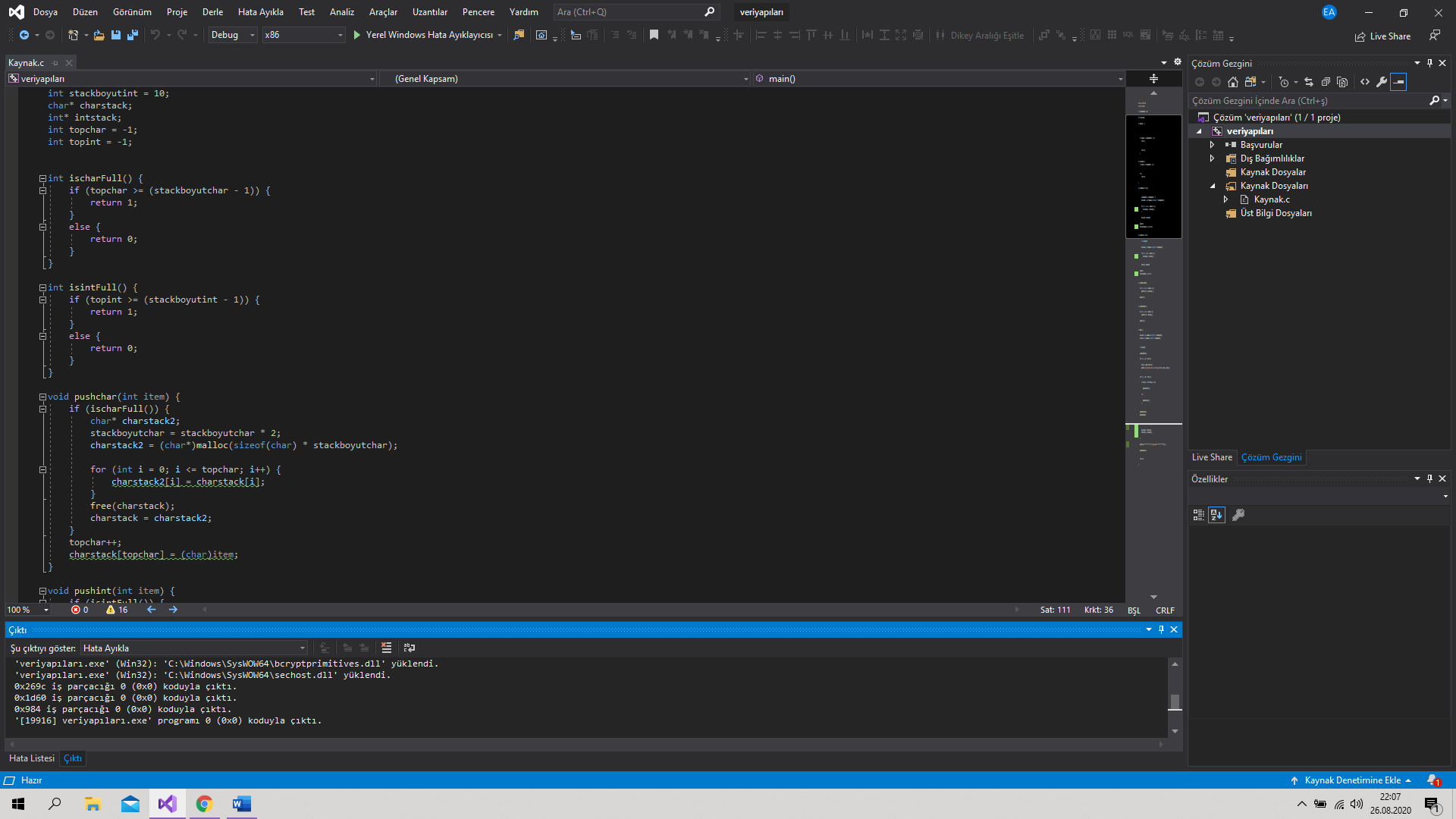
1. Adım: Gelen değerleri atamak için iki tane stack oluşturulur. İşleme devam etmeden önce stack boyutu tanımlanır (benim için 10’du).

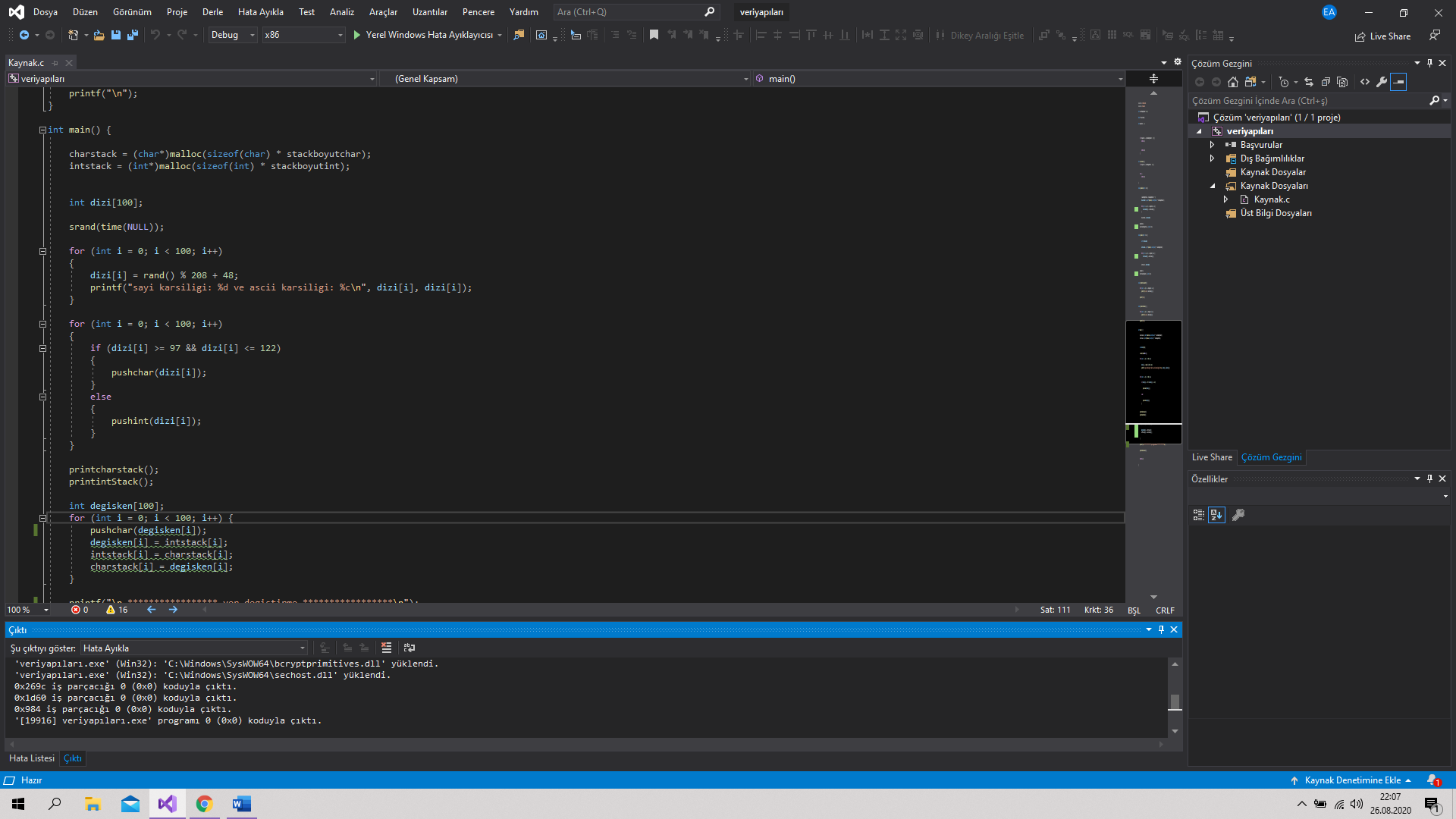


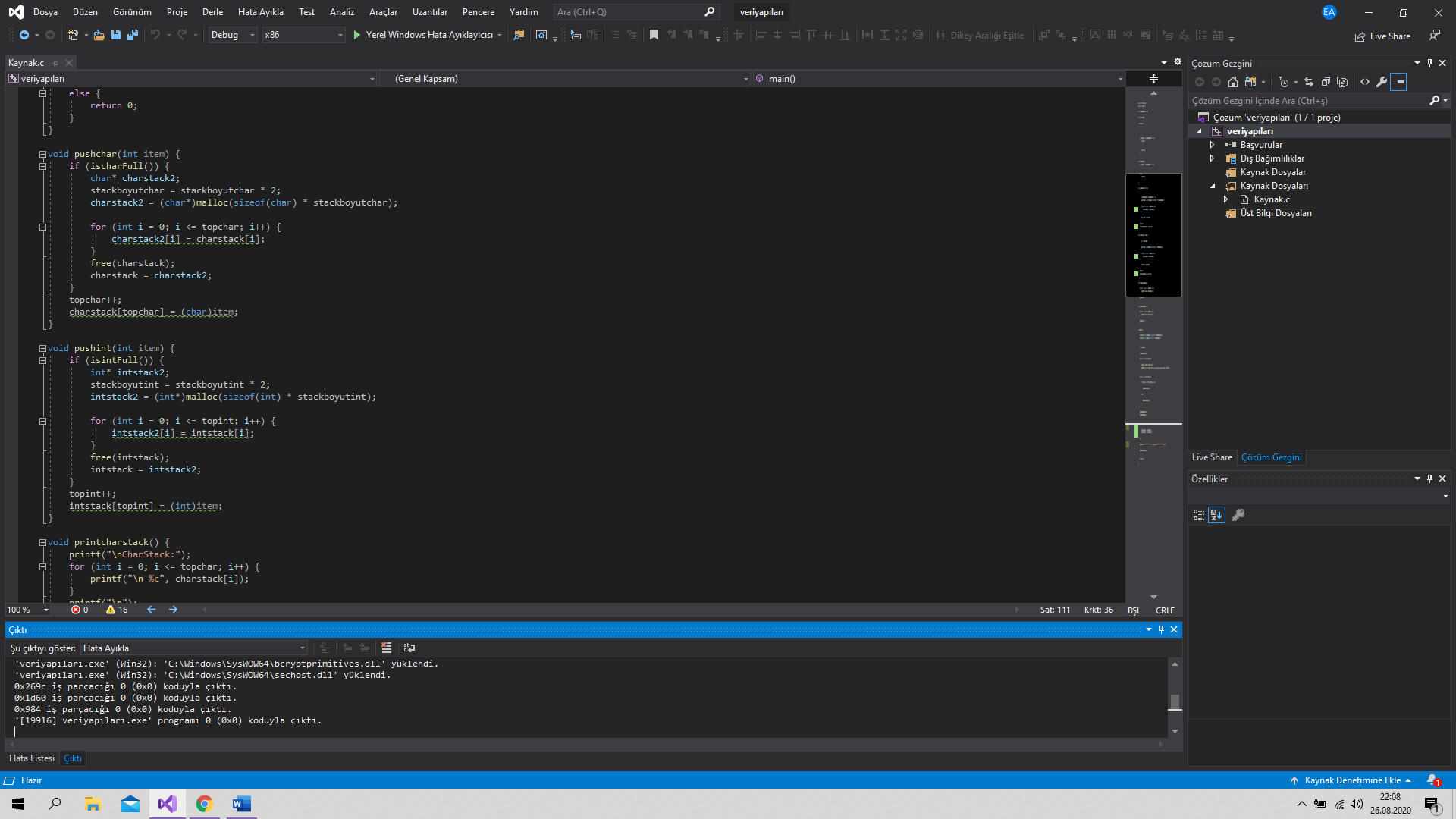
1. Adım: 100 sayı üretilmesine karşı 10 boyutlu bir stack alanı yeterli olmayacağı için malloc fonksiyonu kullanılır. Malloc, stack kapasitesi dolduğunda stack’in boyutunun iki katına çıkmasını sağlayan bir fonksiyondur ve top ile kontrol edilir.

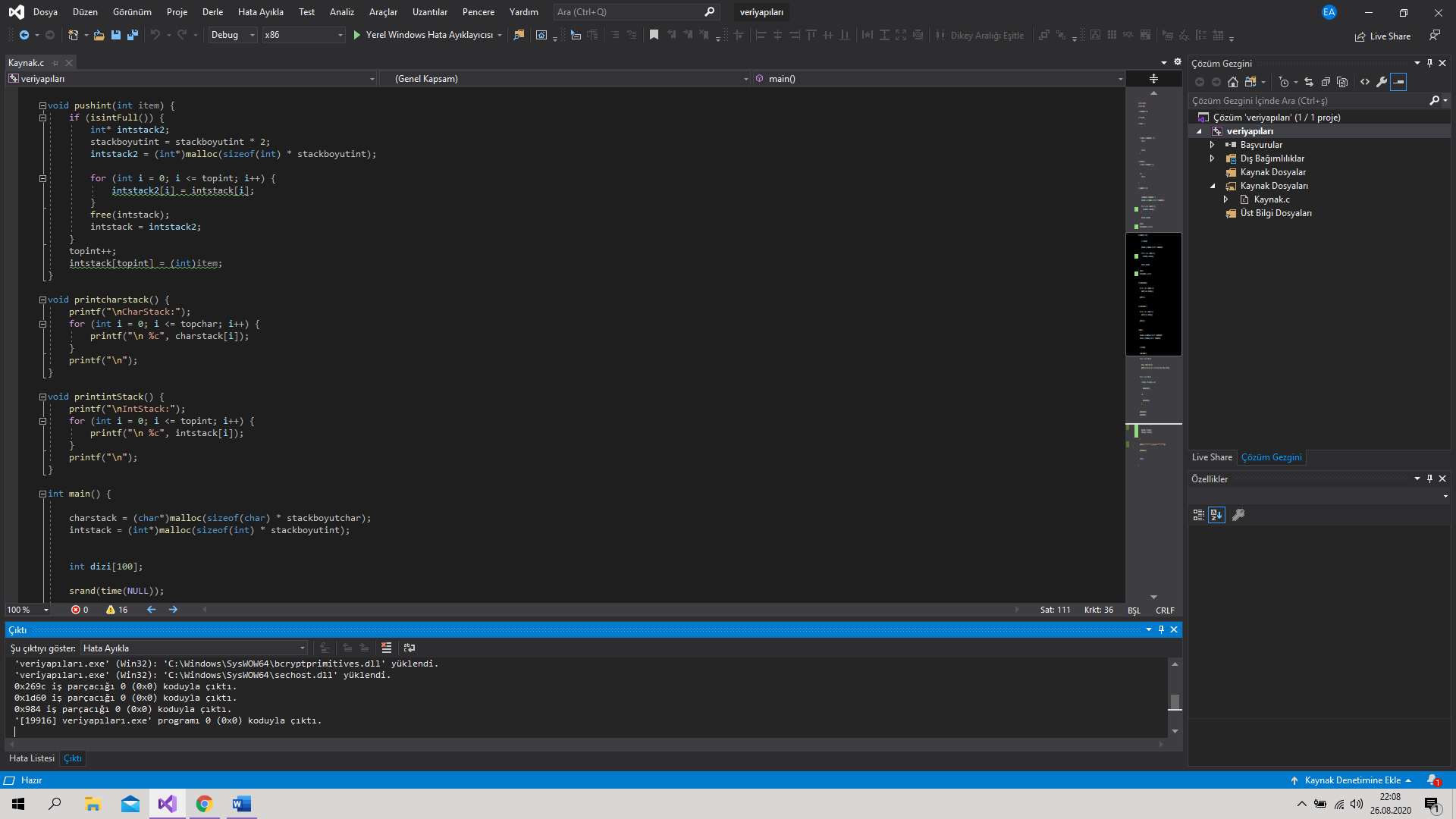


1. Adım: Her bir stack için isFull, Push ve Print fonksiyonları yazılır. isFull, stack’ lerin dolup dolmadığını kontrol eder. Her eleman atama işleminde bu kontrol tekrarlanır. Push, dizi ile üretilen elemanları uygun olan stack’e ekler. Üretilen sayı 97 – 122 arasında ise ascii değeri küçük harftir ve char stack’e gitmelidir. Bu aralığın dışında ise int stack’e gitmelidir. Print ise programın sonunda çıktıları yazdırmamızı sağlar.

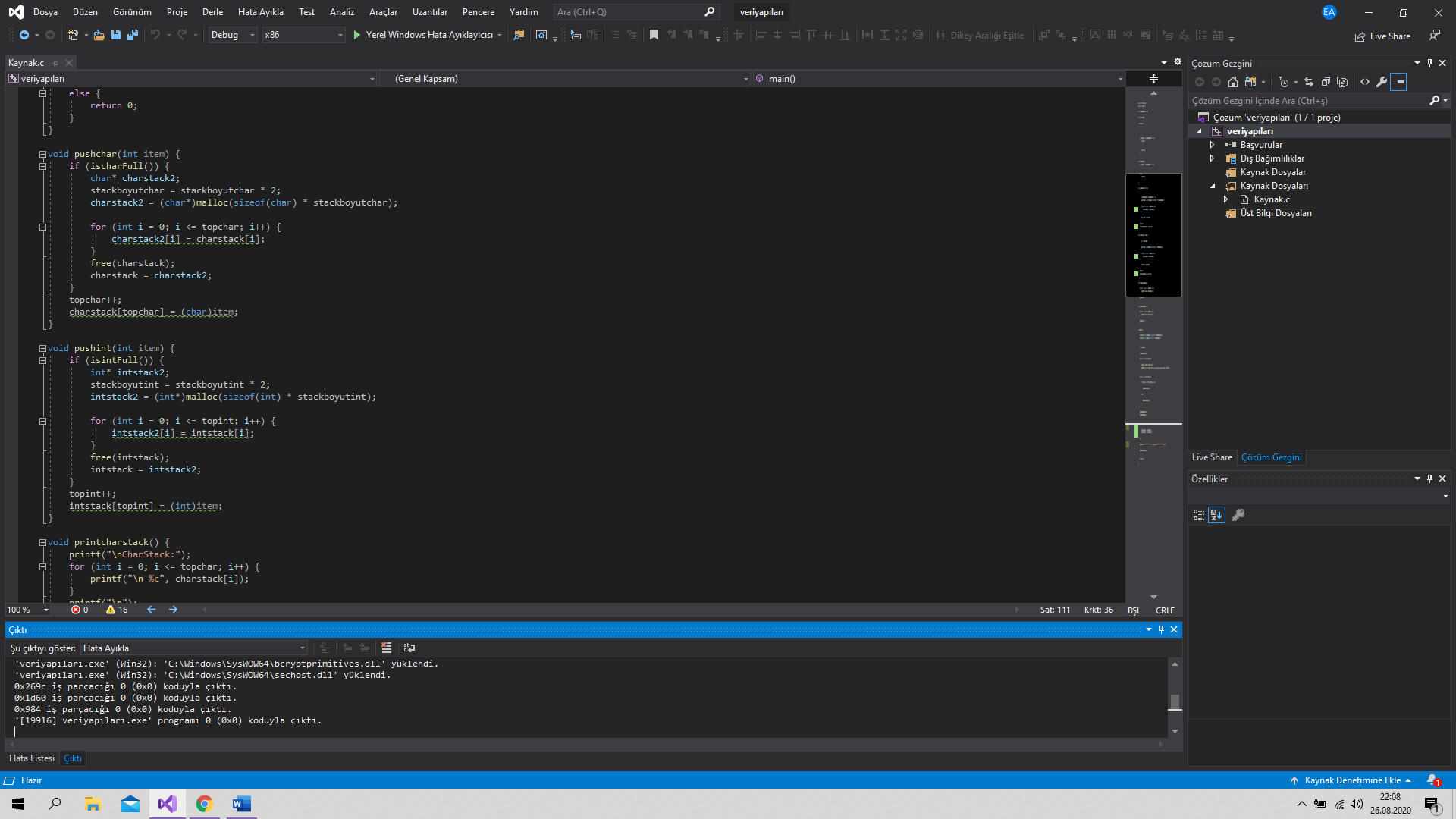




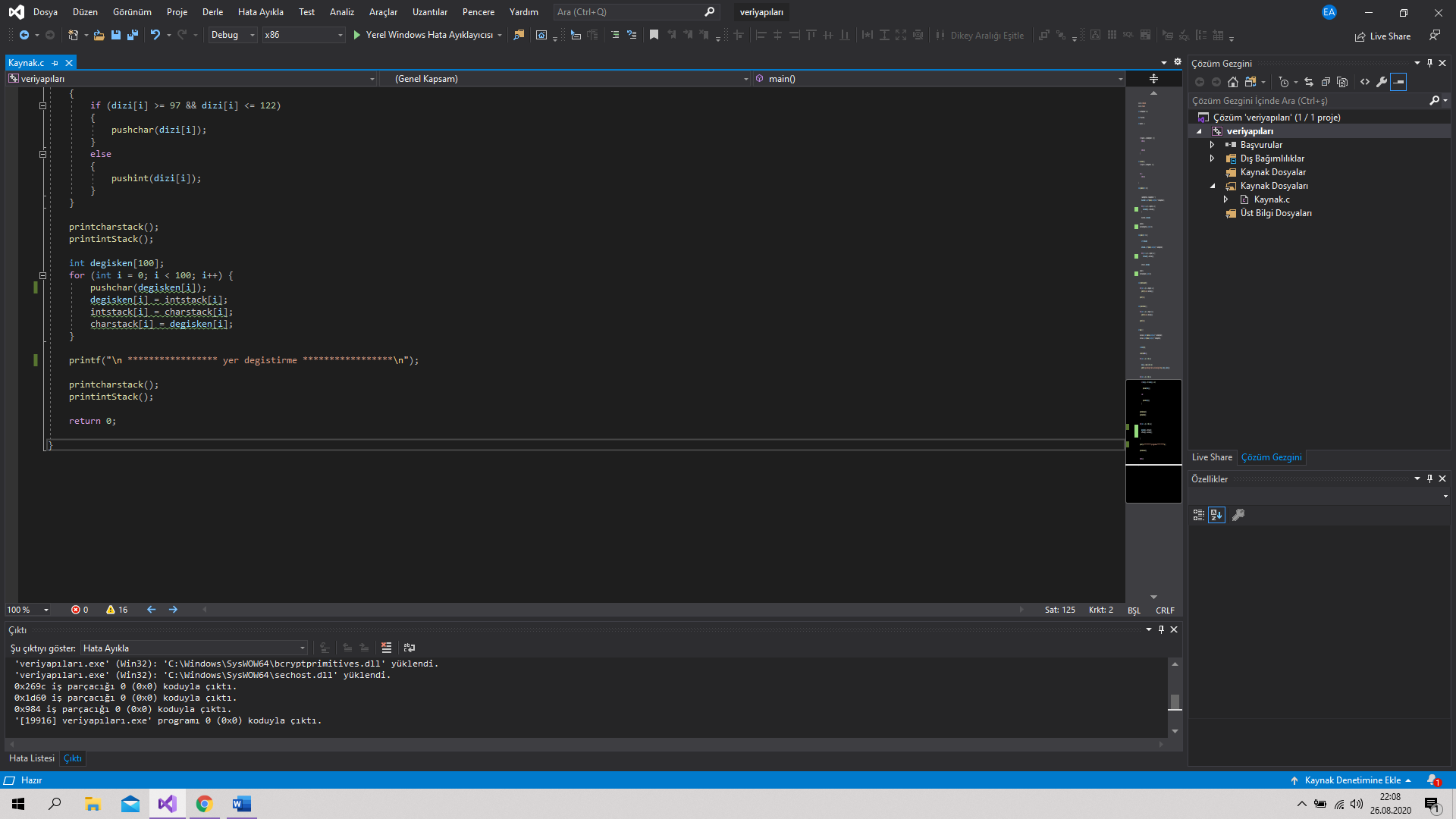




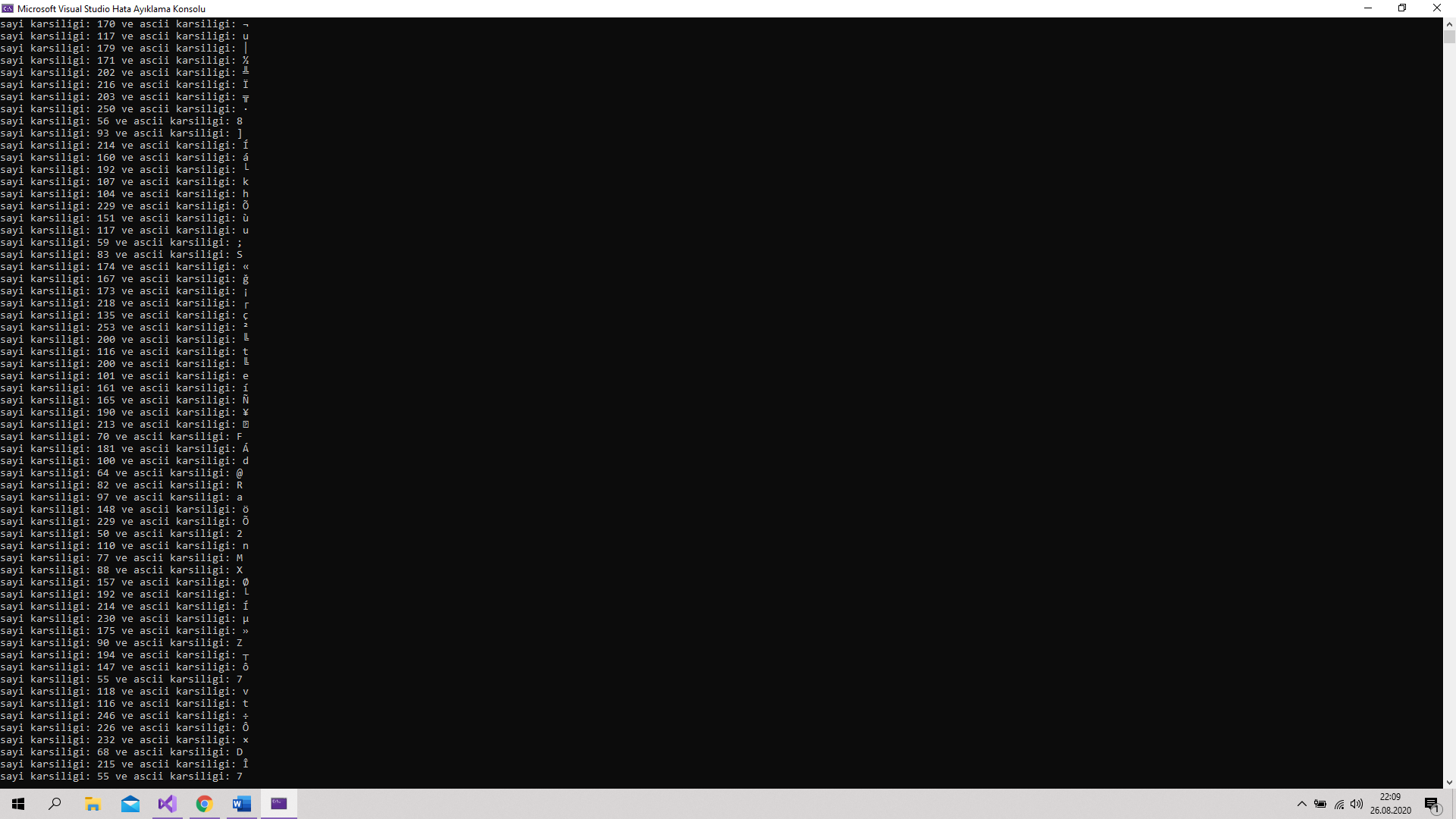
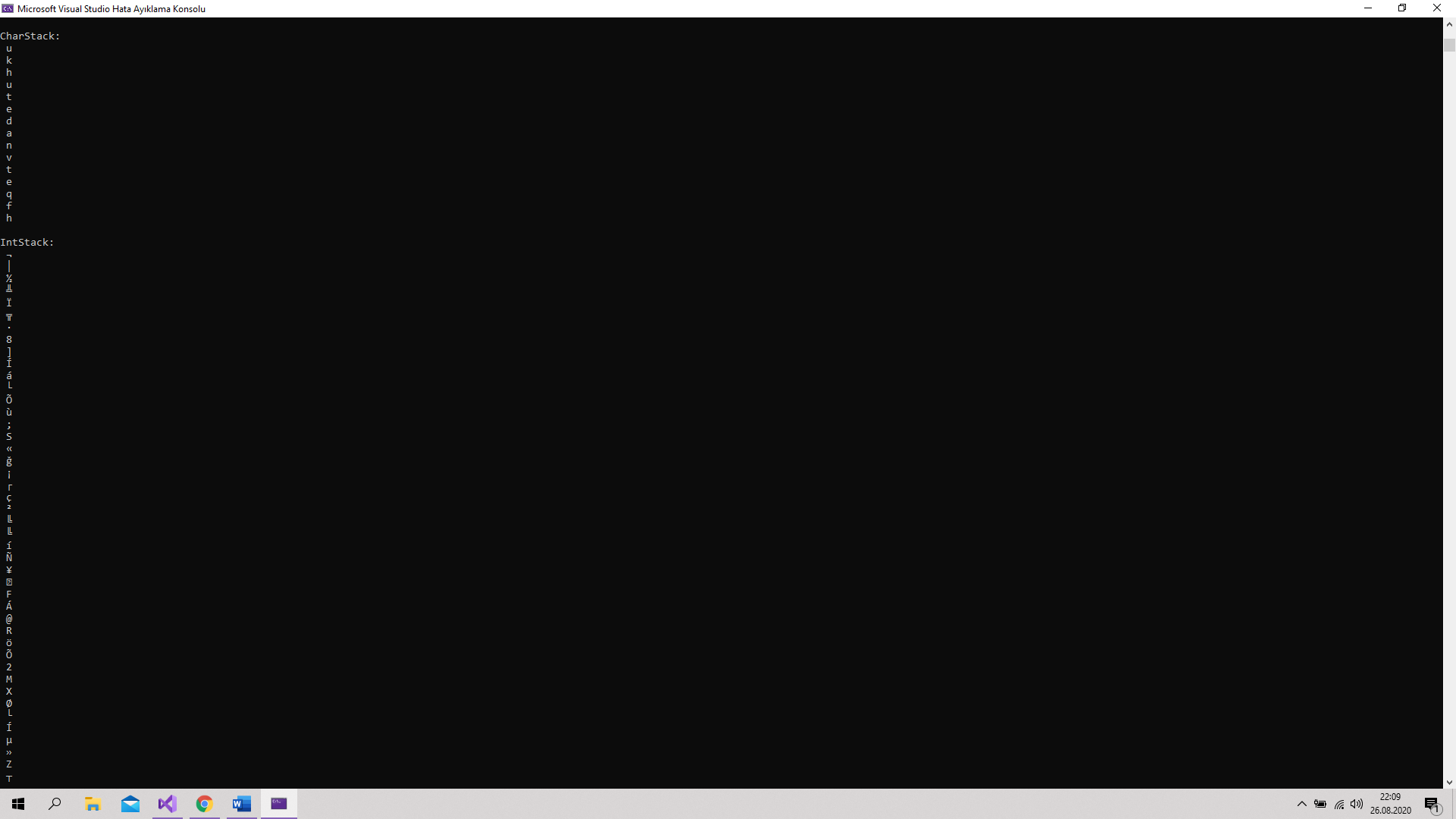
1. Adım: isFull kontrolünde stack boyutu dolmuş ise tekrar malloc fonksiyonunu kullanılıp ikinci bir yedek stack oluşturulur. Elemanlar yeni değerlerle birlikte boyutu artırılmış yedek stack’e gittikten sonra asıl stack sıfırlanır ve yedek stack, asıl stack’e aktarılır.

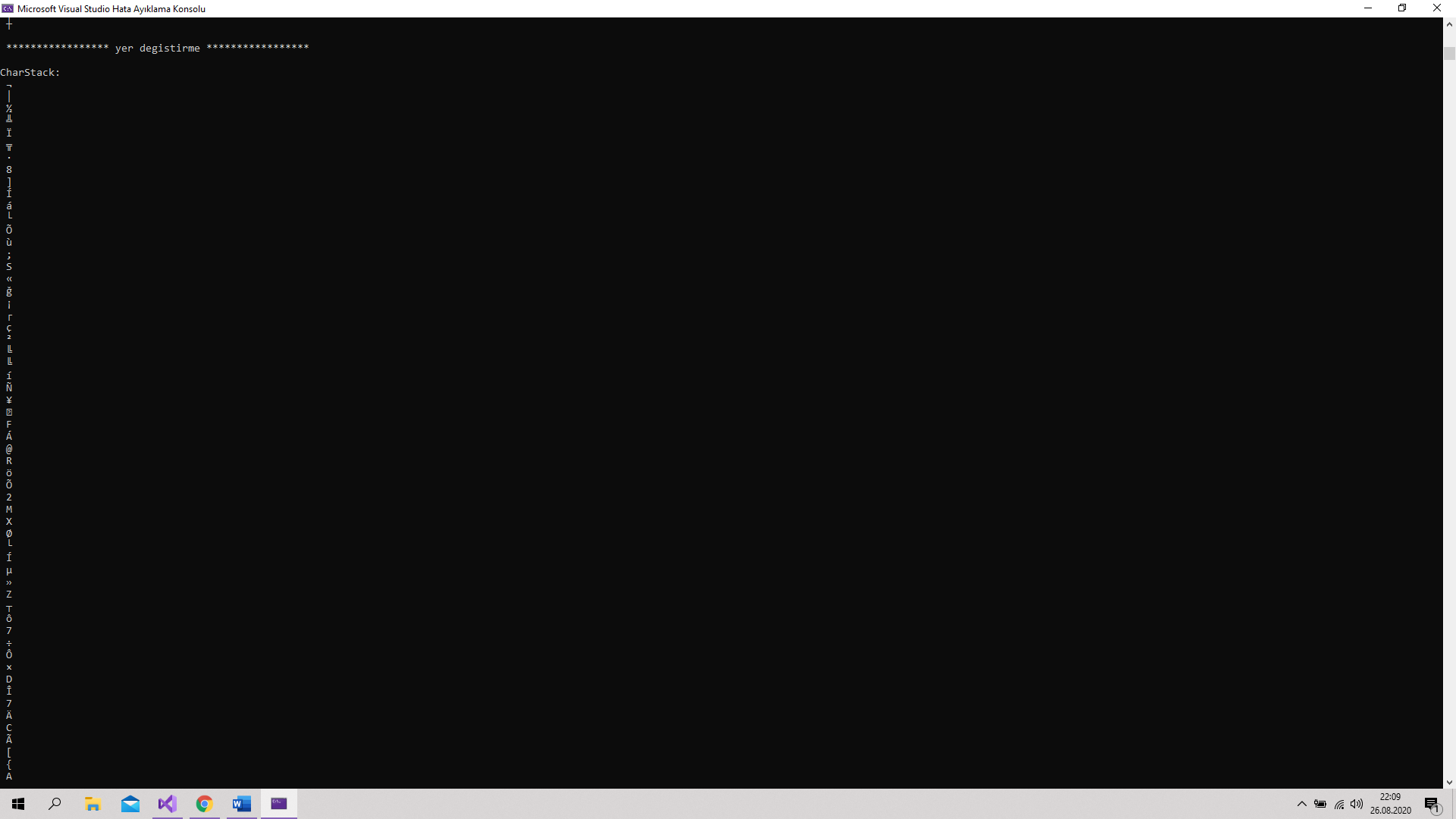
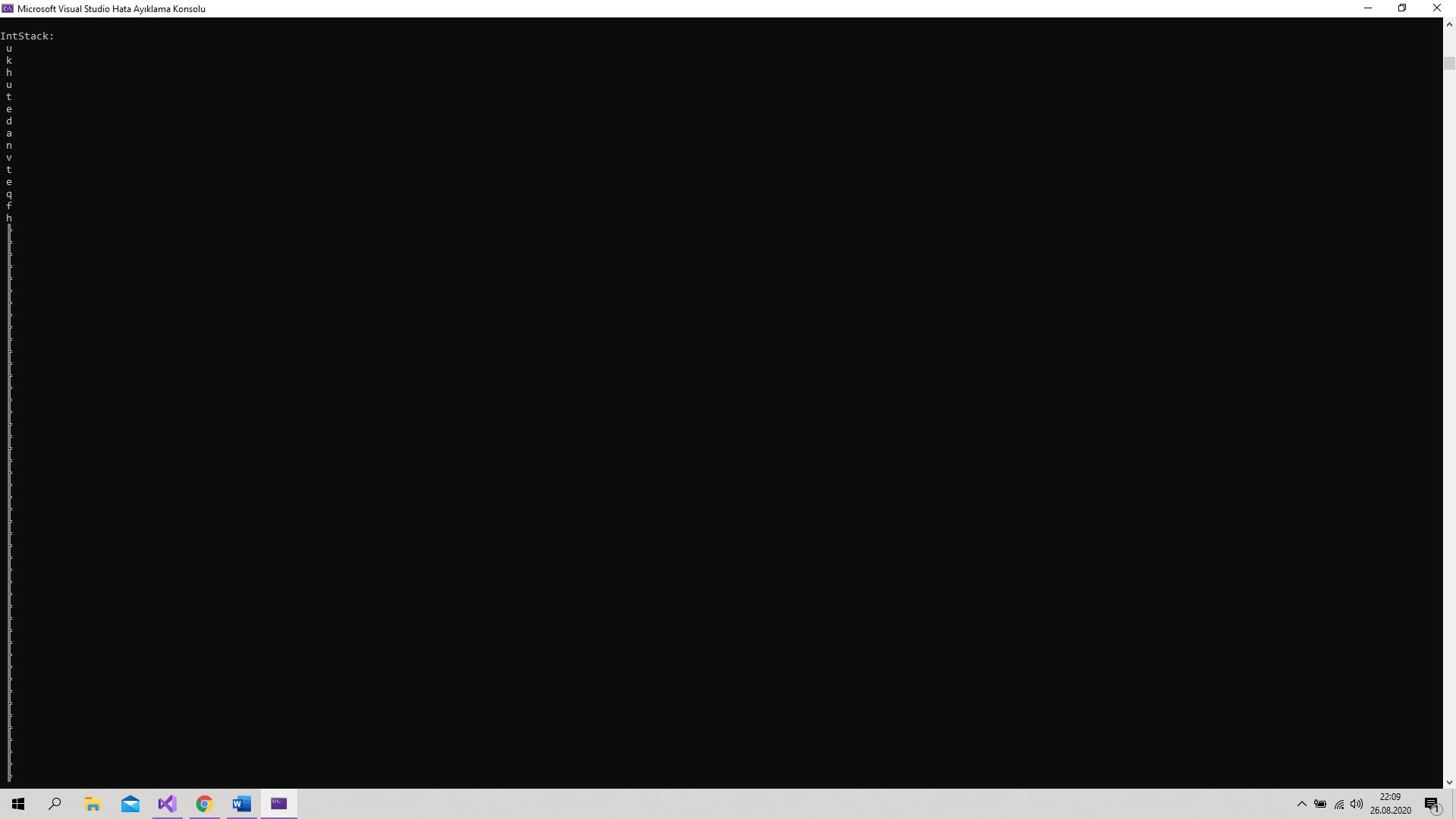


1. Adım: Son adım olarak iki stack’e ait verilerin yer değiştirilmesi için değişken adında yeni bir dizi oluşturulur. Eleman sayısı 100 olarak tanımlanır çünkü program 100 sayı üreterek başladı ve değişken dizide de max ihtiyaç duyabileceği eleman sayısı 100’dür. Bu dizi push fonksiyonu ile çağırılıp gelen char değerleri değişkene atılır, boş kalan char stack’e de int değerleri atılır, int stack’e de char stack atandıktan sonra yer değiştirme tamamlanır.



1. Adım: Program çıktısı ancak değişkeni 100 elemanla tanımladığımız için ve int stack, char stack tüm alanı doldurmadığı için dizi kalan kısma boş karakter atadı.

Projenin Kaynak Kodları

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int stackboyutchar = 10;

int stackboyutint = 10;

char\* charstack;

int\* intstack;

int topchar = -1;

int topint = -1;

int ischarFull() {

if (topchar >= (stackboyutchar - 1)) {

return 1;

}

else {

return 0;

}

}

int isintFull() {

if (topint >= (stackboyutint - 1)) {

return 1;

}

else {

return 0;

}

}

void pushchar(int item) {

if (ischarFull()) {

char\* charstack2;

stackboyutchar = stackboyutchar \* 2;

charstack2 = (char\*)malloc(sizeof(char) \* stackboyutchar);

for (int i = 0; i <= topchar; i++) {

charstack2[i] = charstack[i];

}

free(charstack);

charstack = charstack2;

}

topchar++;

charstack[topchar] = (char)item;

}

void pushint(int item) {

if (isintFull()) {

int\* intstack2;

stackboyutint = stackboyutint \* 2;

intstack2 = (int\*)malloc(sizeof(int) \* stackboyutint);

for (int i = 0; i <= topint; i++) {

intstack2[i] = intstack[i];

}

free(intstack);

intstack = intstack2;

}

topint++;

intstack[topint] = (int)item;

}

void printcharstack() {

printf("\nCharStack:");

for (int i = 0; i <= topchar; i++) {

printf("\n %c", charstack[i]);

}

printf("\n");

}

void printintStack() {

printf("\nIntStack:");

for (int i = 0; i <= topint; i++) {

printf("\n %c", intstack[i]);

}

printf("\n");

}

int main() {

charstack = (char\*)malloc(sizeof(char) \* stackboyutchar);

intstack = (int\*)malloc(sizeof(int) \* stackboyutint);

int dizi[100];

srand(time(NULL));

for (int i = 0; i < 100; i++)

{

dizi[i] = rand() % 208 + 48;

printf("sayi karsiligi: %d ve ascii karsiligi: %c\n", dizi[i], dizi[i]);

}

for (int i = 0; i < 100; i++)

{

if (dizi[i] >= 97 && dizi[i] <= 122)

{

pushchar(dizi[i]);

}

else

{

pushint(dizi[i]);

}

}

printcharstack();

printintStack();

int degisken[100];

for (int i = 0; i < 100; i++) {

pushchar(degisken[i]);

degisken[i] = intstack[i];

intstack[i] = charstack[i];

charstack[i] = degisken[i];

}

printf("\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* yer degistirme \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printcharstack();

printintStack();

return 0;

}