

BURSA TEKNİK ÜNİVERSİTESİ BİLGİSAYAR MÜHENDİSLİĞİ

BLM0111 ALGORİTMALAR VE PROGRAMLAMA DERSİ PROJE ÖDEVİ:

İngilizce mi Almanca mı?

1.Öğrenci Adı : Moussa Bane 2.Öğrenci Adı : Mustafa Erhan Portakal

1.Öğrenci Numarası : 20360859102 2.Öğrenci Numarası : 20360859032

Bir char dizisi tanımladık. Bu dizide öncelikle 100 karekter artı 1 dizi sonu karekteri(\0) ekledik fakat daha çok karekter dizisi girilebilir diye 1000 + 1 karekter sınırı tanımladık.

Daha sonra printf komudu sayesinde Kullanıcı ile iletişime geçtik.

Gets komudu ile Alinan diye adlandırdığımız char dizisini kullanıcıdan aldık.

Taslak kodda tanımlı olan filter\_str() parametresini kullanarak olmaması gereken karekterleri(!,’,^,+…) boşluk karekterleri ile değiştirdik. Tüm büyük harfleri de küçük harflere çevirdik. Bu işlemler için 4 tane iç içe for döngüsü kurduk. Daha sonra ise sonucu printf komudu ile kullanıcıya char dizinin son halini bildirdik.

Taslak kodda tanımlı olan Calculate\_frequencies\_bi() parametresini kullanarak cümlede geçen bigramları hesapladık. Bunu için taslak kodda tanımlı olan matrix\_bigram\_strings[][] char ‘ ını kullandık.

const char matrix\_bigram\_strings[10][2] ={"th", "he", "in", "er", "an", "en", "ch", "de", "ei", "te"};

Bu char dizisindeki elemanları Alinan dediğimiz char dizisinde aratmak için calculated\_frequencies[] float dizisini bir kez daha tanımladık. Maksat unutmayalım diye zaten taslak kodda tanımlanmıştı. Biz kendimiz sadece bu fonksiyon kısmında geçerli olacak sayac diye float veri türü tanımladık. Sonra for döngümüzü açtık ve döngümüzün içinde “i” adında int veri türü tanımladık. Matrix\_bigram\_strings matris dizisinde 10 adet karekter tanımlı olduğu için gereksiz aramayı azaltmak istedik. Strlen() komutu ile kullanıcıdan aldığımız yazının karekter sayısına göre arama yapmasını sağladık yine gereksiz aramayı azaltıp bilgisayarı yormamak için. if komutu ile char veri türünde tanımlı olan matrix\_bigram\_strings matrisindeki karekterleri arattık ve bulunan karekterleri sayaç ‘ a işledik. Daha sonra printf komutu ile kullanıcıyla iletişime geçip metnin üzerindeki bigram frekanslarını yazacağımızı söyledik. for komutu ile bir döngü kurduk ve kullanıcıya matrix\_bigram\_strings matris dizisindeki matrisleri ve bu matrislerde oluşan sayaç değerlerini karşılıklı yazdırdık ve bu fonksiyonu burada bitirdik.

const char matrix\_trigram\_strings[10][3]={"the", "and", "ing", "ent", "ion", "der", "sch", "ich", "nde", "die"};

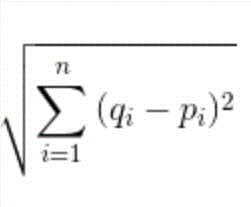
Bigram frekans değerlerini hesapladıktan sonra trigram frekans değerlerini hesaplamak için taslak kodumuzda olan calculate\_frequencies\_tri() parametrisini main fonksiyonumuza işledik. Bu fonksiyonu çağırdık. Bu fonksiyonun içinde int veri türünde i ve j yi tanımladık. Float veri türünde ise yine sayac ‘ ı tanımladık. İç içe döngü tanımladık ve bigramda yaptığımız adımları tekrardan gerçekleştirdik ve calculated\_frequencies[] dizisinin 10. elemandan sonrasına yerleştirdik. Daha sonra ise printf ile kullanıcı ile iletişime geçtik ve matrix\_trigram\_string[][]in ilk elemanından son elemanına kadar yazdık ve bu elemanların cümlemizde ne kadar geçtiğini gösterdik. Burada calculate\_frequencies\_tri fonksiyonumuzu bitirdik.

Sonraki işleme geçmeden önce, kullanıcıdan aldığımız metinin üzerinde hesapladığımız frekanları (hem bigramlar için hem de trigramlar için) ekrana yazdırmak istedik. Bunun için for ve printf komutlarından faydalandık ve main fonksiyonumuza geri döndük.

Main fonksiyonumuza geri döndükten sonra yine taslak kodda tanımlı olan void calculate\_distances() parametresini kullanarak void calculate\_distances() fonksiyonuna geçiş yaptık.

Bu fonksiyonda adından anlaşılacağı gibi taslak kodda İngilizce ve almanca için verilmiş olan frekans değerlerine oklid uzaklığı formülünü kullanarak uzaklık hesabı yaptık.

Oklid Uzaklığı Formulü =>



İlk olarak float veri türünde toplam ve dist değişkenlerini tanımladık ve ekrana yazdıracağımız değerleri %.2f ile virgülden sonra 2 sayı gösterilecek şekilde kullanıcıya gösterdik.

//İngilizce dilinin frekans değerleri

const float frequency\_eng[20]={ 2.71, 2.33, 2.03, 1.78, 1.61, 1.13, 0.01, 0.01, 0.01, 0.01, 1.81, 0.73, 0.72, 0.42, 0.42, 0.01, 0.01, 0.01, 0.01, 0.01 };

//Almanca dilinin frekans değerleri

const float frequency\_germ[20]={ 0.01, 0.89, 1.71, 3.90, 1.07, 3.61, 2.36, 2.31, 1.98, 1.98, 0.01, 0.01, 0.01, 0.01, 0.01, 1.04, 0.76, 0.75, 0.72, 0.62 };

Ondan sonra başka bir for döngüsü ile ilk olarak calculated\_frequencies dizisi ile İngilizce frekans değerleri arasındaki oklid uzaklığını hesapladık ve bu değeri kullanıcıya bildirdik. Daha sonra aynı işlemi for döngüsü sayesinde almanca dilinin frekans değerleri ile uzaklığını hesapladık ve bu değeri kullanıcıya bildirdik. Ayrıca bu iki değeri İngilizce ve almanca sırasıyla taslak kodda tanımlı olan discantes dizisinin 0. Ve 1. Elemanlarına yazdırdık ve fonksiyonu burada bitirdik. Main fonksiyonuna geri döndük.

Son olarak metnin yazıldığı dilin calculate\_distances dizisine göre tahmin edilmesi kaldı. Taslak kodda olan void detect\_lang() parametresini main fonksiyonumuza yazarak son kez main fonksiyonumuzdan çıktık ve detect\_lang fonksiyonuna geçiş yaptık

Bu fonksiyonda if kullanılarak discantes[0] a atadığımız İngilizce için uzaklıkla discantes[1] e atadığımız almanca için uzaklığı karşılaştırdık ve küçük olan değere göre cümlenin hangi dil olduğunu tahmin ettik ve bu dili taslak kodda tanımlı olan languages matrisi sayesinde for döngüsü ile kullanıcıya bitirdik ve fonksiyondan çıkıp main fonksiyonuna döndük ve uygulamayı bitirdik.

Örnek Görüntü Örnekleri=>