**BURSA TEKNİK ÜNİVERSİTESİ**

**BİLGİSAYAR MÜHENDİSLİĞİ**

**ALGORİTMA VE PROGRAMLAMA DERSİ**

**GÜZ DÖNEMİ**

**PROJE ÖDEVİ**

**GİRİLEN METNİN DİLİNİN BELİRLENMESİ**

**HAZIRLAYANLAR:**

**İREM ATAK 20360859074**

**BAHADIR AKSAKAL 20360859079**

**ÖZET**

**10000 karakteri aşmayacak bir metin kullanıcı tarafından girilir.**

**Girilen metin yabancı karakterlerden temizlenir.**

**Bigram ve trigramlarin metinde kaç kez geçtiği hesaplanır.**

**Verilen frekans değerleriyle hesaplama yapılır ve metnin dilinin Almanca mı**

**İngilizce mi olduğu ekrana yazdırılır.**

**YAPILAN ÇALIŞMALAR**

**Adım 1:**

**Metin alınırken kullanılan void metni\_al(char str[]) fonksiyonu gets()**

**kullanımı sonucu oluşabilecek hataları (copy-paste-error) gidermek için**

**getchar() isimli fonksiyonu for döngüsü içinde kullandık. Böylece copy-**

**paste sorununu çözdük. Daha farklı fonksiyonları da denedik ama**

**tekrardan hata aldık. En doğru yöntemin for döngüsünü kullanmak**

**olduğuna karar verdik. Art arda metin girilmesi durumunda her seferinde**

**build and run yapmak yerine maindeki bütün fonksiyonlarımızı sonsuz**

**bir döngünün içine aldık ve bu döngüde bir devam koşulu koyduk .**

**O koşul gerçekleşmediği sürece yeni metin girilebiliyor. Koşul**

**gerçekleştiğindeyse yeni metin girilemiyor .Metnin tamamı girildikten**

**sonra enter ile alt satıra geçip eof karakter tuşlanır ve tekrardan entera**

**tıklanır . Böylece program çalışır.**

**Adım 2:**

**Girilen metin void filter\_str(char str[]) fonksiyonu ile yabancı karakterlerin**

**yerine boşluk karakteri yerleştirilir. Böylece metin yabancı karakterlerden**

**temizlenmiş olur. Metni temizledikten sonra bütün karakterleri küçük harfe**

**dönüştürdük. Amacımız bigram ve trigramlar aranırken sorun çıkmamasıydı.**

**Adım 3:**

**Girilen metinde void calculate\_frequencies\_bi(char str[]) fonksiyonu ile**

**her bir bigramın metinde kaç kez tekrar ettiğini hesapadık. Şöyle bir sorunla**

**karşılaştık. Örneğin “the” kelimesinde hem “th” bigramı hem de “he” bigramı**

**var. İlk yazdığımız fonksiyonda ilk bigramı saydıktan sonra ikinci bigramı**

**saymıyordu. Yani bigramları eksik hesaplıyordu. Projede yazdığımız**

**fonksiyonla bu sorunu çözdük ve minimum düzeyde RAM kullanımına dikkat**

**ettik. Hesaplanan frekansaları calculated\_frequencies[i] dizisine sırasıyla**

**yerleştirdik.**

**Adım 4:**

**Girilen metinde void calculate\_frequencies\_tri(char str[]) fonksiyonu ile**

**her bir trigramın metinde kaç kez tekrar ettiğini hesapladık. Hesaplanan**

**frekansaları 10. elemandan başlayacak şekilde calculated\_frequencies[i+10]**

**dizisine sırasıyla yerleştirdik.**

**Adım 5:**

**void calculate\_distances() fonksiyonu ile İngilizce ve Almanca dilleri için ayrı ayrı puanlama yaptık . Distances[0] dizisine İngilizce için hesaplanan puan**

**atanır. distances[1] dizisineyse Almanca için hesaplanan puan atanır.**

**Projede bizden öklit hesaplaması yapmamız isteniyordu ama bu**

**hesaplama kısa metinlerde hatalı sonuç veriyordu. Özellikle 3 satırı geçmeyen**

**metinlerde ciddi sorunlar oluyordu. Doğruluk oranı nerdeyse %70’in**

**altındaydı. Bu sorunu çözecek yeni bir yöntem geliştirdik. Yöntemimiz şu**

**şekilde frekansı 1’den yüksek olan bigram ve trigramları seçtik. İngilizce için**

**bunların tekrar sayısını toplayıp distances[0] dizisine yerleştirdik. Almanca**

**içinse aynı şekilde toplayıp distances[1] dizisine yerleştirdik.**

**Adım 6:**

**void detect\_lang() fonksiyonunda distances[0] ve distances[1]’in değerlerini**

**karşılaştırdık . Distances dizisinin 0. indeksteki değeri daha büyükse metnin**

**dilinin İngilizce olduğu ekrana yazdırılıyor, değilse Almanca olduğu**

**yazdırılıyor. Belirlenen dilin doğruluğunu arttırmak için bir algoritma**

**kullandık. Bir sayaç ekledik. Bu sayaç tekrar sıklığı 1’den az olan bigram ve**

**trigramları sayıyor. Eğer sayaç 20 ise girilen metin çok kısadır ve sonuç doğru**

**olmayabilir. Bunu kullanıcıya da belirtmek için uyarı mesajı gönderdik. 2. bir**

**sayaç ekledik. Bu sayaç bigram ve trigramların tekrar sıklığı 5’den az**

**olanlarını sayıyor. 2. sayaç 20 ise girilen metin muhtemelen bir cümledir.**

**Metinse de farklı bir dil olabilir. Bunu belirtmek için de kullanıcıya bir uyarı**

**mesajı gönderiyoruz. Projemiz geliştirilmeye açık. Farklı dilde daha uzun bir**

**metin girildiğinde sayac1 ve sayac2 işlevsiz kalacaktır.**

**NOT 1:**

**Bizden bigram ve trigramların frekanslarının hesaplanması istendi.**

**Frekansları ,bigram ve trigramların tekrar sayılarını toplam kelime sayısına**

**bölüp hesaplamayı planladık. void metinde\_kelime\_sayisi\_bulucu(char str[])**

**fonksiyonunu metindeki toplam kelimeyi bulmak için kullandık. Bu**

**fonksiyonun çalışma mantığı şöyledir: 2 boşluk arasındaki char dizisini kelime**

**olarak tanımlar ve bundan yola çıkarak metindeki kelime sayısını hesaplar. Bu**

**yöntemin her zaman doğru sonuç vermediğini fark ettik ve yöntemimizi**

**değiştirdik. Bu fonksiyonu herhangi bir yerde kullanmadık. Fonksiyonun**

**projemizde bulunmasını istedik çünkü proje geliştirilirse ihtiyaç**

**duyulabileceğini düşündük.**

**NOT 2:**

**void SetColor(int ForgC) bu fonksiyonun yerine ilk başta System(“COLOR D”);**

**fonksiyonunu kullanarak bütün metnin rengini değiştirebiliyorduk ama**

**istediğimiz yerlerde değişiklik yapamıyorduk. İstediğimiz kısımlarda metnin**

**rengini değiştirebileceğimiz bir fonksiyon bulmak istedik . Yeterli bilgimiz**

**olmadığından çeşitli forumlardan yardım alarak bu fonksiyonu bulduk.**

**Fonksiyonun kütüphanelerini ve fonksiyonu metnimize ekleyerek**

**istediğimiz renklendirme işlemlerini yaptık. Bu fonksiyonu kullanmamızın**

**nedeni kullanıcının dikkatini çekmek ve görsel bir estetiklik katmaktır.**

**Projenin işlevi kadar tasarımının da önemli olduğunu düşünüyoruz.**