# Kelime Tahmini Uygulaması

## Giriş

Uygulama kullanıcının aklından tuttuğu rastgele bir kelimeyi tahmin etmeye çalışmaktadır. En fazla 12 tahminde tutulan kelimenin bulunması hedeflenmiştir. İlk tahmin hariç her yapılan tahminde kullanıcı tuttuğu gizli kelime ile yapılan tahmindeki ortak harflerin sayısı ile ilgili bilgiyi uygulamaya vermektedir. Bir sonraki tahmin bu bilgiye dayanarak daha isabetli yapılmaktadır.

## Tahmin Sistemi

### Algoritma

1. Bir sözlük yüklenir.
2. Mevcut sözlükten rastgele bir tahmin yapılır ve bu tahmin sözlükten çıkartılır.
3. Tahmin doğruysa sonlanır.
4. Kullanıcıdan daha önceden belirlenmiş bir puanlama sistemine göre bu tahmine puan vermesi istenilir.
5. Aynı puanlama sistemine göre sözlükteki tüm kelimeler mevcut tahminle kıyaslanır ve puan üretilir.
6. Sözlükteki kelimelerden kullanıcının vermiş olduğu puan ile aynı puanı üretemeyen kelimeler çıkartılır.
7. Sözlükte kalan kelime sayısı birden küçükse “kelime bulunamadı” çıktısı üretilir ve sonlanır.
8. Sözlükte kalan kelimelerden rastgele yeni bir tahmin yapılır ve bu tahmin sözlükten çıkartılır.
9. Tahmin doğru değilse 3. adıma dönülür.
10. Sonlanır.

### Tahminin Yapıldığı Metodun Kodu

Kod C++ dilinde .Net framework’ü kullanılarak yazılmıştır.

Program için WordPrediction adında bir class yazılmıştır.  
Üye değişkenlerden array<System::String^>^ dictionary programa yüklenen tüm kelimelerin listesidir.   
Üye değişken array<System::String^>^ remaining tüm kelimelerin listesinin, doğru kelime olma ihtimali olmayan kelimelerinin "zzz" metni ile değiştirilmiş olan halidir.   
Üye değişken int remainingCount mevcut sözlükteki, doğru kelime olma ihtimali olan kelimelerin sayısıdır.   
Üye değişken int tried kaç deneme yapıldığının sayısıdır.   
Üye değişken int guessIndex mevcut tahminin mevcut sözlükteki indisidir

Tahminin yapıldığı üye fonksiyonun kodu aşağıdaki gibidir.

System::String^ WordPrediction::MakeGuess(int pointNormal, int pointCartesian)

{

if (tried == 0) {

tried++;

System::Random^ rand = gcnew System::Random();

this->guessIndex = rand->Next(remaining->Length);

this->remaining[guessIndex] = "zzz";

this->remainingCount--;

return this->dictionary[this->guessIndex];

}

else {

tried++;

System::String^ guessWord = this->dictionary[this->guessIndex];

System::String^ currWord;

int currPointC;

int currPointN;

for (int i = 0; i < this->remaining->Length; i++) {

currWord = this->remaining[i];

if (currWord != "zzz") {

currPointC = CompareCartesian(currWord, guessWord);

currPointN = CompareNormal(currWord, guessWord);

if (currPointC != pointCartesian || currPointN != pointNormal) {

this->remaining[i] = "zzz";

this->remainingCount--;

}

}

}

if (this->remainingCount < 1) {

this->guessIndex = -1;

return "Word not found";

}

else {

System::Random^ rand = gcnew System::Random;

do

{

this->guessIndex = rand->Next(remaining->Length);

} while (this->remaining[this->guessIndex] == "zzz");

this->remaining[guessIndex] = "zzz";

this->remainingCount--;

return this->dictionary[this->guessIndex];

}

}

}

## Puanlama Sistemi

### Normal Mod

#### Algoritma

1. “kıyaslanan” kelime ve “tahmin” kelimesi alınır.
2. “puan” değerine 0 verilir.
3. “i” değerine 0 verilir.
4. “j” değerine 0 verilir.
5. Eğer “kıyaslanan[i]” ile “tahmin[j]” aynı harf ise “puan” değeri 1 artırılır ve 8. adıma atlanır.
6. “j” değeri 1 artırılır.
7. “j”, “tahmin” kelimesinin boyundan küçükse 5. adıma atlanır.
8. “i” değeri 1 artırılır.
9. “i”, “kıyaslanan” kelimenin boyundan küçükse 4. adıma atlanır.
10. “puan” değeri çıktı olarak verilir ve sonlanır.

#### Kod

int CompareNormal(System::String ^ str1, System::String ^ str2)

{

int point = 0;

for (int i = 0; i < str1->Length; i++) {

for (int j = 0; j < str2->Length; j++)

{

if (str1[i] == str2[j]) {

point++;

break;

}

}

}

return point;

}

### Kartezyen Mod

#### Algoritma

1. “kıyaslanan” kelime ve “tahmin” kelimesi alınır.
2. “puan” değerine 0 verilir.
3. “i” değerine 0 verilir.
4. “j” değerine 0 verilir.
5. Eğer “kıyaslanan[i]” ile “tahmin[j]” aynı harf ise “puan” değeri 1 artırılır.
6. “j” değeri 1 artırılır.
7. “j”, “tahmin” kelimesinin boyundan küçükse 5. adıma atlanır.
8. “i” değeri 1 artırılır.
9. “i”, “kıyaslanan” kelimenin boyundan küçükse 4. adıma atlanır.
10. “puan” değeri çıktı olarak verilir ve sonlanır.

#### Kod

int CompareCartesian(System::String ^ str1, System::String ^ str2)

{

int point = 0;

for (int i = 0; i < str1->Length; i++) {

for (int j = 0; j < str2->Length; j++)

{

if (str1[i] == str2[j]) {

point++;

}

}

}

return point;

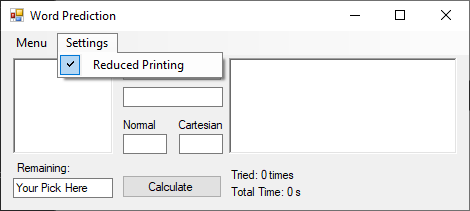
}

## Programın Kullanımı

### Ayarlar

“Settings” menüsündeki “Reduced Printing” seçili halde değilse mevcut sözlükteki tüm kelimeler sol panelde yazdırılır. Mevcut kelime sayısının 1000’in üzerinde olduğu durumlarda bu işlem kullanıcıyı rahatsız edebilecek şekilde uzun sürmektedir.

“Settings” menüsündeki “Reduced Printing” seçili hale getirilirse mevcut kelime sayısı 1000’in üzerinde olduğu durumlarda “More than 1000 words” mesajı yazdırılmaktadır. Mevcut kelime sayısının 1000’in altında olduğu durumlarda kelimelerin tamamı yazdırılmaktadır.



### Sözlük Yükleme