İnsan Asmaca Oyunu

Problem 1:

Bu ödevde klasik kelime oyunu olan İnsan Asmaca'nın bir varyasyonunu kodlayacaksınız. Oyunun kurallarını bilmiyorsanız, https://www.wikiwand.com/tr/Adam_asmaca adresini okuyabilirsiniz. Bu problem sizi korkutmasın, aslında göründüğünden çok daha kolay! Siz asıl oyunu uygulamadan önce yardımcı fonksiyonların yaratılmasında size rehberlik ederek bu sorunu basitleştireceğiz.

A.Başlangıç

"insan_asmaca.py" ve "tdk_sozcukler2.csv" dosyalarını indirin ve ikisini de bilgisayarınızda aynı dizine kaydedin. Dosyalarınızın doğru kaydedildiğinden emin olmak için herhangi bir kod yazmadan önce insan_asmaca.py dosyasını çalıştırın. Size verdiğimiz kod, bir dosyadan kelimelerle yüklenir. Ekranda çıktı olarak aşağıdakileri göreceksiniz:

```
1.  Dosyadan kelime listesi okunuyor...
2.  213358 kelimelik liste hazırlandı.
```

Yukarıdaki metni görürseniz İnsan Asmaca Oyun Gereksinimleri'ne devam edin. Bunu yapmazsanız, her iki dosyanın da aynı yere kaydedilip kaydedilmediğini iki kez kontrol edin!

B.İnsan Asmaca Oyun Gereksinimleri

İnsan Asmaca adında, kullanıcının bilgisayara karşı insan asmaca oynamasına izin verecek bir fonksiyon uygulayacaksınız. Bilgisayar kelimeyi seçer ve oyuncu kelimedeki harfleri tahmin etmeye çalışır.

İşte uygulamak istediğimiz kodun genel davranışı. Gözünüzü korkutmayın! Bu sadece bir tanımdır; bunu adımlara ayıracağız ve daha sonra problem setinde daha fazla işlevsel özellikler sunacağız, bu yüzden okumaya devam edin!

1.Bilgisayar, tdk_sozcukler2.csv dosyasında sağlanan mevcut sözcükler listesinden rastgele bir sözcük seçmelidir. tdk_sozcukler2.csv dosyasının tümü küçük harflerle yazılmış sözcükler içerdiğine dikkat edin.

2. Kullanıcıya başlangıçta belirli sayıda tahmin hakkı verilir.

3. Oyun etkileşimlidir, kullanıcı tahminlerini ve bilgisayar aşağıdakilerden birini girer:

a. gizli kelimede varsa harfi ortaya çıkarır

b. kullanıcıyı cezalandırır ve kalan tahmin sayısını günceller

4. Oyun, kullanıcı gizli kelimeyi tahmin ettiğinde veya kullanıcının tahminleri bittiğinde sona erer.

Problem 2:

İnsan Asmaca Bölüm 1: Yardımcı Fonksiyonlar

İnsan asmaca oyununu organize etmek için kod yazmadan önce, sorunu mantıksal alt görevlere ayıracağız, bu oyunun çalışması için sahip olmanız gereken üç yardımcı işlev oluşturacağız. Bu, hesaplamalı problem çözme için yaygın bir yaklaşımdır ve deneyimlemeye başlamanızı istiyoruz.

insan_asmaca.py dosyası, çözümünüzü yazarken kullanabileceğiniz bir dizi önceden uygulanmış fonksiyona sahiptir. Her bir yardımcı işlevi nasıl kullanacağınızı docstrings okuyarak anlamanız gerekse de, dosyanın en üstündeki iki işlevdeki kodu göz ardı edebilirsiniz.

1A) Kelimenin tahmin edilip edilmediğini belirleyin

İlk olarak, iki parametre alan *kelime_tahmin_edildi_mi* fonksiyonunu uygulayın, *gizli_kelime* ve bir harf listesi (dizeler) olan *tahmin_edilen_harfler*. Bu fonksiyon, eğer *gizli_kelime* tahmin edildiyse, bir boolean True döndürür (yani, *gizli_kelime*'nin tüm harfleri *tahmin_edilen_harfler*), aksi takdirde False döndürür. Bu fonksiyon, insan asmaca oyununun ne zaman başarıyla tamamlandığına karar vermenize yardımcı olacak ve harfleri gizli kelimeye göre kontrol eden herhangi bir yinelemeli döngü için bir son test haline gelecektir.

Bu fonksiyon için, *gizli_kelime* ve *tahmin_edilen_harfler* için tüm harflerin küçük harf olduğunu varsayabilirsiniz.

Örnek kullanım:

```
>>> gizli_kelime = 'elma'
>>> tahmin_edilen_harfler = ['e', 'i', 'k', 'p', 'r', 's']
>>> print(kelime_tahmin_edildi_mi(gizli_kelime, tahmin_edilen_harfler))
False
```

1B) Kullanıcıdan tahmin alma

Daha sonra, iki parametre alan *tahmin_edilen_kelimeyi_al* fonksiyonunu uygulayın: parametreler *gizli_kelime* ve bir harf listesi olan, *tahmin_edilen_harfler*. Bu fonksiyon, *gizli_kelime*'de *tahmin_edilen_harfler*'i esas alarak, harfler ve alt çizgilerden oluşan bir dize döndürür. Bu, *kelime_tahmin_edildi_mi*'den çok farklı olmamalıdır!

Bilinmeyen harfleri temsil etmek için bir alt çizgi(_) ve ardından bir boşluk kullanacağız. Başka semboller seçebilirdik, ancak alt çizgi ve boşluk kombinasyonu görülebilir ve kolayca ayırt edilebilir. Boşluğun çok önemli olduğuna dikkat edin, aksi takdirde ____ 'nin dört öğe uzunluğunda mı yoksa üç öğe mi olduğunu ayırt etmek zordur. Buna **kullanılabilirlik** denir, programlama sırasında programınızın kullanılabilirliğini göz önünde bulundurmak çok önemlidir. Kullanıcılar programınızı anlamakta veya çalıştırmakta zorlanırsa, onu kullanmazlar! Programınızı tasarlarken kullanılabilirliği düşünmenizi öneririz.

İpucu: Fonksiyonunuz tasarlarken, tamamlandığında hangi bilgileri geri vermek istediğinizi, bir veri yapısı üzerinde döngü oluştururken bu bilgileri depolamak için bir yere ihtiyacınız olup olmadığını ve birikmiş sonucunuza nasıl bilgi eklemek istediğinizi düşünün.

1C) Tüm uygun harflerin alınması

Daha sonra, bir parametrede harflerin bir listesini alan, parametresi *tahmin_edilen_harfler* olan *uygun_harfleri_al* fonksiyonunu uygulayın. Bu fonksiyon, tüm harflerle ifade edilmeyen küçük Türkçe harflerden oluşan bir dize döndürür.

Bu fonksiyon, harfleri alfabetik sırada döndürmelidir. Bu fonksiyon için, tahmin_edilen_harfler'deki tüm harflerin küçük harf olduğunu varsayabilirsiniz.

İpucu: Tüm küçük harflerden oluşan bir dize olan *alfabe* değişkenini kullanmayı düşünebilirsiniz:

```
>>> print(alfabe)
abcçdefgğhıijklmnoöprsştuüvyz
```

Örnek kullanım:

```
>>> tahmin_edilen_harfler = ['e', 'i', 'k', 'p', 'r', 's']
>>> print uygun_harfleri_al(tahmin_edilen_harfler)
abcçdfgğhıjlmnoötuüvyz
```

Problem 3:

Artık bazı yararlı fonksiyonlar oluşturduğunuza göre, kullanıcının tahmin etmesi gereken *gizli_kelime*'nin bir parametresini alan insan asmaca fonksiyonunu uygulamaya dönebilirsiniz. Başlangıçta, bu fonksiyonu çalıştırdığınızda bu gizli kelimeyi manuel olarak ayarlayabilirsiniz (ve yapmalısınız!) - bu, kodunuzu test etmeyi kolaylaştıracaktır. Ama sonunda, sizi veya başka bir kullanıcıyı bu işlevi çalıştırarak oyunu oynamaya davet etmeden önce bilgisayarın bu gizli kelimeyi rastgele seçmesini isteyeceksiniz.

İnsan asmaca fonksiyonunu çağırmak, kullanıcı ve bilgisayar arasında etkileşimli bir İnsan Asmaca oyunu başlatır. Kodunuzu tasarlarken, önceki bölümde tanımladığınız üç yardımcı fonksiyondan (*kelime_tahmin_edildi_mi*, *tahmin_edilen_kelimeyi_al* ve *uygun_harfleri_al*) yararlandığınızdan emin olun!

Aşağıda farklı kategorilere ayrılmış oyun gereksinimleri verilmiştir. Uygulamanızın tüm gereksinimleri karşıladığından emin olun!

A.Oyun Mimarisi

1. Bilgisayar, *tdk_sozcukler2.csv* dosyasında sağlanan kullanılabilir sözcükler listesinden rastgele bir sözcük seçmelidir. Kelime listesini yükleme ve rastgele bir kelime seçme fonksiyonları, *insan_asmaca.py*'de sizin için zaten sağlanmıştır.

2. Kullanıcılar 6 tahminle başlar.

3. Oyunun başında, kullanıcının bilgisayarın gösterdiği kelimenin kaç harf içerdiğini ve kaç tahminle başladığını bilmesini sağlayın.

4. Bilgisayar, kullanıcının şu ana kadar tahmin etmediği tüm harfleri takip eder ve her turdan önce kullanıcıya "kalan harfleri" gösterir.

Örnek oyun implementasyonu:

```
Dosyadan kelime listesi okunuyor...
213318 kelimelik liste hazırlandı.
İnsan Asmaca oyununa hoşgeldiniz!
4 harfli bir kelime düşünüyorum.
-------------
6 tahmin hakkınız kaldı.
Uygun harfler: abcçdefgğhıijklmnoöpqrsştuüvyz
```

B.Kullanıcı-Bilgisayar Etkileşimi

Oyun etkileşimli olmalı ve aşağıdaki gibi işlemelidr:

1. Her tahminden önce, kullanıcıya şunları göstermelisiniz:

a. Kullanıcıya her tahminden sonra kaç tahmin kaldığını hatırlatın.

b. Kullanıcının henüz tahmin etmediği tüm harfleri gösterin.

2. Kullanıcıdan her seferinde bir tahmin yapmasını isteyin. (Kullanıcıdan ne tür girdiler bekleyebileceğinizi görmek için aşağıdaki kullanıcı girişi gereksinimlerine bakabilirsiniz)

3. Her tahminden hemen sonra kullanıcıya, tahmin ettiği harfin bilgisayarın kelimesinde olup olmadığı söylenmelidir.

4. Her tahminden sonra, kullanıcıya bilgisayarın kelimesini, tahmin edilen harflerle ve alt çizgi ve boşluk (_) ile de tahmin edilmemiş harfleri göstermelisiniz.

5. Tahminin sonunda, tek tek tahminleri birbirinden ayırmaya yardımcı olması için birkaç tire işareti (-----) yazdırın.

Örnek Oyun Uygulaması:

(Aşağıdaki mavi renk, bilgisayarın çıktısının aksine, yalnızca kullanıcının ne yazdığını size göstermek için yazılmıştır.)

```
6 tahmin hakkınız kaldı.
Uygun harfler: abcçdefgğhıijklmnoöpqrsştuüvyz
Bir harf tahmin edin: a
Güzel tahmin: _ a_ _
------------
6 tahmin hakkınız kaldı.
Uygun harfler: bcçdefgğhıijklmnoöpqrsştuüvyz
Bir harf tahmin edin: b
Oops! Bu harf benim kelimemde yok: _ a_ _
```

C.Kullanıcı Girdi Gereksinimleri

1. Kullanıcının her seferinde yalnızca bir karakter tahmin edeceğini varsayabilirsiniz, ancak kullanıcı herhangi bir sayı, sembol veya harf seçebilir. Kodunuz büyük ve küçük harfleri geçerli tahminler olarak kabul etmelidir!

2. Kullanıcı alfabe dışında bir şey girerse (semboller, sayılar), kullanıcıya yalnızca bir alfabe girebileceğini söyleyin. Kullanıcı bunu yanlışlıkla yapabileceği için oyunun başında 3 uyarı alması gerekir. Geçersiz bir giriş veya önceden tahmin ettikleri bir harf girdiklerinde, bir uyarı kaybetmeleri gerekir. Kullanıcının hiçbir uyarısı kalmamışsa ve geçersiz bir giriş yaparsa, bir tahminde bulunamamalıdır.

İpucu 1: Kullanıcının tahminini almak için *input()* işlevine yapılan çağrıları kullanın.

a. Kullanıcı girişinin bir alfabe olup olmadığını kontrol edin

b. Kullanıcı bir büyük veya küçük harf girmiyorsa, uyarı ve tahmin haklarından bir uyarı veya bir tahmin çıkarın.

İpucu 2: Dize fonksiyonları olan *str.isalpha("abcçde")* ve *str.lower("abcçde")* işinize yarayabilir. Bunlar daha ayrıntılı olarak *help(str.alpha)* veya *help(str.lower)* ile dokümantasyona göz atabilirsiniz.

İpucu 3: Veritabanından alınan kelime listesi küçük harflerden oluşacağından, kullanıcı girdisini her defasında küçük harfe dönüştürmek iyi bir fikir olabilir.

Örnek oyun uygulaması:

```
3 uyarı hakkınız kaldı..
6 tahmin hakkınız kaldı..
Uygun harfler: bcçdefgğhıijklmnoöpqrsştuüvyz
Bir harf tahmin edin: s
Oops! Bu harf benim kelimemde yok: _ a_ _
------------
5 tahmin hakkınız kaldı.
Available letters: bcdefghijklmnopqrtuvwxyz
Bir harf tahmin edin: $
Oops! Bu geçerli bir harf değil. 2 uyarı hakkınız kaldı: _ a_ _
```

D.Oyun Kuralları

1. Kullanıcı 3 uyarı ile başlar.

2. Kullanıcı bir alfabe dışında herhangi bir şey girerse (semboller, sayılar), kullanıcıya yalnızca bir alfabe girebileceğini söyleyin.

a. Kullanıcının bir veya daha fazla uyarısı kaldıysa, kullanıcı bir uyarıyı kaybetmelidir. Kullanıcıya kalan uyarı hakkı sayısını söyleyin.

b. Kullanıcının kalan uyarı hakkı yoksa, bir tahmin hakkı kaybetmesi gerekir.

3. Kullanıcı önceden tahmin edilmiş bir harf girerse, kullanıcıya harfin önceden tahmin edildiğini bildiren bir mesaj yazdırın.

a. Kullanıcının bir veya daha fazla uyarısı kaldıysa, kullanıcı bir uyarıyı kaybetmelidir. Kullanıcıya kalan uyarı sayısını söyleyin.

b. Kullanıcının hiçbir uyarısı yoksa, bir tahmin hakkını kaybetmesi gerekir.

4. Kullanıcı daha önce tahmin edilmeyen bir harf girerse ve harf gizli kelimede mevcut ise, kullanıcının tahmin hakkı azalmaz.

5. **Ünsüzler**: Kullanıcı tahmin edilmemiş bir ünsüz harf girerse ve ünsüz harf gizli kelimede mevcut değilse, ve bu bir ünsüz harf ise kullanıcı **bir** tahmin hakkı kaybeder.

6. **Ünlüler**: Sesli harf tahmin edilmemişse ve sesli harf gizli kelimede mevcut değilse, kullanıcı **iki** tahmin hakkı kaybeder. Ünlüler a, e, ı, i, o, ö, u ve ü şeklindedir. y harfi sesli olarak sayılmaz.

Örnek oyun uygulaması:

```
5 tahmin hakkınız kaldı.
Uygun harfler: bcdefghijklmnopqrtuvwxyz
Bir harf tahmin edin: t
Güzel tahmin: ta_ t
------------
5 tahmin hakkınız kaldı.
Uygun harfler: bcdefghijklmnopqrtuvwxyz
Bir harf tahmin edin: e
Oops! Bu harf benim kelimemde yok: ta_ t
------------
3 tahmin hakkınız kaldı.
Uygun harfler: bcdfghijklmnopqrtuvwxyz
Bir harf tahmin edin: e
Oops! Bu harfi zaten tahmin etmiştiniz. Şimdi 2 uyarı hakkınız kaldı:
ta_ t
```

E.Oyunun Sonlanması

1. Kullanıcı tam kelimeyi oluşturduğunda veya tahminler bittiğinde oyun sona ermelidir.

2. Kullanıcının kelimeyi tamamlamadan önce tahmin hakkı tükenirse, ona kaybettiğini söyleyin ve oyun bittiğinde kelimeyi kullanıcıya açıklayın.

3. Kullanıcı kazanırsa, bir tebrik mesajı yazdırın ve kullanıcıya puanını bildirin.

4. Toplam puan, kullanıcının *gizli_kelime*'yi tahmin ettikten sonra, *gizli_kelime*'deki benzersiz(unique) harflerin sayısı ile kalan tahminlerin sayısının çarpımıdır.

Toplam puan = kalan_tahmin_sayısı * gizli_kelime'deki benzersiz harf sayısı.

Örnek uygulama:

```
3 tahmin hakkınız kaldı.
Uygun harfler: bcçdefgğhıijklmnoöpuüvyz
Bir harf tahmin edin: r
Güzel tahmin: tart
------------
Tebrikler, kazandınız!
Bu oyun için toplam puanınız: 9
```

F.Genel İpuçları

1. İhtiyacınız olursa ek yardımcı işlevler yazmayı düşünün.

2. Saklamak isteyebileceğiniz dört önemli bilgi vardır:

a. *gizli_kelime*: Tahmin edilecek kelime. Bu, insan_asmaca fonksiyonu için parametre adı olarak zaten kullanılıyor.

b. *tahmin_edilen_harfler*: Şimdiye kadar tahmin edilen harfler. Kullanıcılar zaten *tahmin_edilen_harfler* içinde olan bir harfi tahmin ederlerse, bunu zaten tahmin ettiklerini ancak bunun için onları cezalandırmadığınızı söyleyen bir mesaj yazdırmalısınız.

c. *kalan_tahminler*: Kullanıcının kalan tahmin hakkı sayısı. Örnek oyunumuzda, yanlış sesli harf seçmenin cezası, yanlış ünsüz seçmenin cezasından farklıdır.

d. *kalan_uyarılar*: Kullanıcının kalan uyarı hakkı sayısı. Bir kullanıcının yalnızca önceden tahmin edilmiş bir sembolü veya bir harfi girdiğinde bir uyarı hakkı kaybettiğini unutmayın.

Kodunuzu tamamlayıp test ettikten sonra (burada "gizli" kelimeyi el ile sağladığınızdan, kodunuzda hata ayıklamanıza yardımcı olduğunu bildiğiniz için), artık bilgisayara karşı oynamayı deneyebilirsiniz. Paylaştığımız dosyanın en altına kaydırırsanız, *if __name__ == "__main__"* metnin altında yorumlanmış iki satır görürsünüz:

```
#gizli_kelime = kelime_seç(kelime_listesi)
#insan_asmaca(gizli_kelime, alfabe=TÜRKÇE_ALFABE)
```

Bu satırlar, incelemek isteyebileceğiniz, paylaştığımız (*insan_asmaca.py*'nin üst kısmına yakın) işlevleri kullanır. Bu satırlar için "yorum işaretlerini kaldırmayı" ve kodunuzu yeniden yüklemeyi deneyin. Bu size, büyük bir kelime kümesini yüklemek ve ardından rastgele birini seçmek için yazdığınız fonksiyonları kullanan bilgisayara karşı kelime tahmin becerinizi deneme şansı verecektir.

Problem 4:

İnsan Asmaca Bölüm 3: İpuçlarıyla Oyun

Bilgisayara karşı İnsan Asmaca oynamayı denediyseniz, özellikle ezoterik bir kelime seçtiğinde ("ezoterik" gibi!) bilgisayarı yenmenin her zaman kolay olmadığını fark etmiş olabilirsiniz. Bilgisayardan, şu anda tahmin ettiğinizle eşleşen tüm kelimelerin listesi gibi bir ipucu sormanız iyi olabilir.

Örneğin, gizli kelime "dokunsal" ise ve şu ana kadar "d" harfini tahmin ettiyseniz, çözümün "d_ _ _ _ _ _ l" olduğunu bilirsiniz, burada beş eksik harfi tahmin etmeniz gerekebilir, eşleşen kelimeler kümesinin (en azından bilgisayarın başlangıçta yüklediği şeye bağlı olarak):

dokunak, dökünük, dokunma

Burada İnsan Asmaca'nın bir varyasyonunu yaratacağız (buna *ipuçlarıyla_insan_asmaca* diyoruz ve onu yazmak için bir başlangıç kodu zaten sağlamıştık). Burada özel karakteri * tahmin ederseniz, bilgisayar

kelime listesindeki mevcut tahmin etmeye çalıştığınız kelimeyle eşleşen tüm kelimeleri bulacaktır ve her birini yazdıracaktır. Elbette, bunu ilk adımda denemenizi önermiyoruz, çünkü bu, yüklediğimiz 213.358 kelimenin tamamını yazdıracaktır! Ancak bir yanıta yaklaşıyorsanız ve tahminleriniz tükeniyorsa, bu yardımcı olabilir.

Bunu yapmak için, sizden önce iki yardımcı işlevi tamamlamanızı isteyeceğiz:

3a)Mevcutta tahmin edilmeye çalışılan kelimeyi eşleştirme

boşluklarla_eşleştir() fonksiyonu iki parametre alır: *benim_kelimem* ve *diğer_kelime*. *benim_kelimem*, tahmin edilen bir kelimenin bir örneğidir, başka bir deyişle, yerlerinde bazı _'ler olabilir ("e_ _ a" gibi). *diğer_kelime* normal bir Türkçe kelimedir.

benim_kelimem 'in tahmin edilen harfleri *diğer_kelime*'nin karşılık gelen harfleriyle eşleşiyorsa bu fonksiyon *True* döndürmelidir. İki kelime aynı uzunlukta değilse veya *benim_kelimem* 'deki tahmin edilen bir harf *diğer_kelime*'deki karşılık gelen karakterle eşleşmiyorsa *False* döndürmelidir.

```
1.  >>> boşluklarla_eşleştir("k_ _ _ uz", "rahat")
2.  False
3.  >>> boşluklarla_eşleştir("k_ _ _ uz", "karpuz")
4.  True
5.  >>> boşluklarla_eşleştir("k_ _ puz", "karpuz")
6.  False
```

İpucu: Uzunlukları karşılaştırmak amacıyla sözcükteki boşluklardan kurtulmak için *strip()* kullanmak isteyebilirsiniz.

3b) Tüm olası eşleşmeleri gösterme

olası_eşleşmeleri_göster fonksiyonu tek bir parametre alır: tahmin edilmeye çalışılan kelimenin bir örneği olan *benim_kelimem*, başka bir deyişle, içinde bazı _'ler olabilir ("e_ _ a" gibi).

Bu fonksiyon, *benim_kelimem* ile eşleşen kelime listesindeki tüm kelimeleri (bunu dosyanın başında tanımladığımıza dikkat edin, satır 53) yazdırmalıdır. Eşleşme yoksa "Eşleşme bulunamadı" yazmalıdır.

Örnek kullanım:

```
1.  >>> olası_eşleştirmeleri_göster("k_ _ _ _ z")
2.  yarpuz
3.  >>> olası_eşleştirmeleri_göster("abbbb_ ")
4.  Eşleşme bulunamadı
```

3c) İpuçlarıyla İnsan Asmaca

Artık insan asmaca için yazdığınız kodu, *ipuçlarıyla_insan_asmaca* fonksiyonunun gövdesi olarak kopyalayabilmeniz, ardından kullanıcının bir yıldız işaretini (*) tahmin edebileceği duruma izin vermek için küçük bir ekleme yapmanız gerekir; bu durumda bilgisayar, tüm bu tahminle eşleşen kelimeleri çıktı olarak verir.

Tahmin yıldız işareti ise, kullanıcı bir tahminde bulunmamalıdır.

Artık orijinal insan_asmaca oyununda yorum satırlarını kaldırdığınız aşağıdaki satırları yorum satırına alabilirsiniz:

```
1. gizli_kelime = kelime_seç(kelime_listesi)
2. insan_asmaca(gizli_kelime, alfabe=TÜRKÇE_ALFABE)
```

Aşağıdaki satırlarında yorum satırlaını kaldırıp, yeni ipuçlarıyla insan asmaca oyununu oynayabilirsiniz:

```
1. #gizli_kelime = kelime_seç(kelime_listesi)
2. #ipuçlarıyla_insan_asmaca(gizli_kelime, alfabe=TÜRKÇE_ALFABE)
```