Cmpe 150 Bonus Lab: Genel Tavsiyeler ve Sık Karşılaşılan Hatalar

Problem Tarifi

- Problemin tarifini dikkatlice okuyun. Soruyu net bir şekilde anlamadığınız sürece çözmeniz de mümkün olmayacak.

- Şunu kabul edebilirsiniz (assume that …) tarzı şeyleri ayrıca sizin kontrol etmenize lüzum yok. Onlar sizin işinizi kolaylaştırmak için veriliyor. Size fazladan yük çıkarmak için değil.

Başlangıç Kodunu İnceleyin

- Biz sizden verdiğimiz fonksiyonun içini yazmanızı istiyoruz. Fakat o fonksiyon haricindeki kod da gayet anlaşılabilir. O kısmı da incelemekten çekinmeyin.

- İncelediğinizde göreceksiniz ki temelde
  - 1) Gerekli inputları kullanıcıdan okuyoruz.
  - 2) Sizin dolduracağınız fonksiyonu o inputları kullanarak çağırıyoruz.
  - 3) Fonksiyonunuzun döndüğü (return ettiği) değeri print ediyoruz.

Kodlamadan Önce Düşünün

- Problemi çözmek için takip edeceğiniz yol (Algoritma) kafanızda net değilse onu başka bir insana tarif edemezsiniz. Benzer şekilde onu bilgisayara da tarif edemezsiniz.

- BAZEN BASİT BİR ÇÖZÜM DE SORUYU GAYET GÜZEL ÇÖZEBİLİR. AYRICA BİLGİSAYARLAR HIZLIDIR VE YORULMAZLAR BU NOKTAYI KULLANMAKTAN ÇEKİNMEYİN.

Hizalama ve Parametreler

- Hizalama kod yazarken (özellikle Python'da) son derece önemli. Her ne zaman ":" kullanılırsa (ÖZELLİKLE def … DEYİP FONKSİYON TANIMLADIKTAN SONRA) takip eden satır ya da satırların (duruma göre) tab tuşu (caps lock'un üstünde) kullanılarak 4 boşluk kaymış halde olması lazım. Her bir hizanın hem hangi satırda başladığı hem de nerede bittiği son derece kritik çünkü kod ona göre çalışacak. Fonksiyon içindeki BÜTÜN satırlardan önce 4 boşluk (tab) bırakılmalı.

- Bir şey fonksiyonun parametresi olarak verildiyse onu tekrar okumaya çalışmayın lütfen. Onu kullanıcıdan okuyup parametre olarak fonksiyona vermek bizim sorumluluğumuz.

Fonksiyonlar

- Bir fonksiyonun içerisinde değişken olarak sadece parametre olarak ne verildiyse onları kullanabiliriz. Daha önce tanımlı herhangi bir değişken eğer parametre olarak fonksiyona verilmediyse onu asla kullanmıyoruz. İsmi parametrelerden birinin ismi ile aynı olsa bile. Zaten kullanmamıza da ihtiyaç yok demektir. Fonksiyon içinde yeni bir değişken tanımlayabiliriz ancak o da kullanıcıdan bir input okuyorsa %99 yanlış yoldayız demektir.

Fonksiyonlar 2

- “return” ve “print” farklı şeylerdir. return satırı çalışınca fonksiyon doğrudan durur ve kodun çalışması nereden çağrıldıysa o kısma döner ve ilgili değeri iletir. Bir değeri dönün (return edin) diyorsak onu print etMEyin. Çünkü döndüğünüz değeri print etmeyi biz zaten fonksiyonu çağırdıktan sonra yapıyoruzdur. Kod çalışıyor mu diye bakmak için sizin kodunuzu çalıştırınca ekrana basılanlar ile bizim beklediklerimiz aynı mı diye string comparison yapıyoruz. O sebeple de fazladan herhangi bir şey print etmeniz halinde çözüm yanlış oluyor.

Ara Aşamalarda Kodu Test Etmek

- Çözümünüzün tamamını bir anda baştan sona yazmak kolay olmayabilir. Çözümü parçalara ayırıp ilerlemek daha sağlıklı bir yoldur.

- Her bir parça için de yeni bir fonksiyon kullanılabilir ya da en azından o parçaya kadarki kısım doğru çalışıyor mu diye print tarzı bir şey kullanılabilir.

For Döngüleri

- if tarzı bir şey kullanmadan return'ü for loop'un içerisine eklersek, daha ilk seferde o kısım çalışır ve bir değer döner. Çoğu zaman for bittikten sonra dönüş yapmak isteriz. O sebeple de yine hizalamayı ayarlamamız lazım.

- Döngü içerisinde bir değişken tanımlarsak her turda o ifade de tekrar çalışacağı için tüm değerleri toplayarak ilerlediğimiz bir değişken olarak onu kullanmamız mümkün değil. Onu döngüden önce tanımlamalıyız.

Değişken İsimlendirmesi

- Değişken türleri ya da fonksiyon isimleri değişken ismi olarak seçilemez:

  int, float, str, list, dict, set, tuple, print, input

- Fakat aşağıdaki gibi küçük farklı hallerini seçersek bir sorun yaşanmaz.

  INT or Float or my_str or List or dict1 or …

None

- Kodu çalıştırdığınızda output'ta "None" içeren bir şey varsa ya da "TypeError: unsupported operand type(s) for +: 'NoneType' and 'int' " gibi bir hata ile karşılaşıyorsanız sebep neredeyse hep return yapmayan bir fonksiyondan gelen ifade (ki gelmiyor) print ya da toplama tarzı bir işlemde kullanılıyor olmaktır. Eğer bahsi geçen fonksiyon sizin yazdığınız bir fonksiyon ise uygun değeri return etmek sorunu çözer.

- HATA MESAJLARINI OKUYUN. ONLAR SİZE YARDIMCI OLSUN DİYE GÖSTERİLİYOR. EKRANDA SADECE "HATA" DENMESİNDENSE BİR İZAH YAPMAYA ÇALIŞIYOR. ŞU SATIRDA ŞÖYLE BİR ŞEY OLDU GİBİ. HATA VEREN SATIR HANGİSİ KONTROL EDİN.

Tester

- KODUN TESTER'DAKİ ÖRNEKLERİ GEÇMESİ VERİLEN HER INPUT İÇİN ÇALIŞACAĞI MANASINA GELMEZ. Main.py'ı kullanarak başka örnekler ile de kodu deneyin.

- Bazen yazdığınız kodu adım adım bilgisayar gibi aklınızda ya da kalem kağıt ile çalıştırmayı deneyin. Konuları daha iyi anlamak için faydalı olacaktır.

Birden Çok Fonksiyon Tanımlama

- Asıl fonksiyonunuzda kullanacağınız yeni fonksiyonlar tanımlamak elbette makul bir ihtiyaç fakat yeni yazdığınız fonksiyonların da doğru çalışıp çalışmadığını sınavda main.py (Main.py değil) gibi yerleri kullanarak test edin lütfen.

- Ek olarak, # satırları dışını değiştirmeden birden çok fonksiyon tanımlama işlemi yaptığınıza emin olun.

Yeteneklerinizi Geliştirmek İçin

- Pratik yapın

- İleride bu alana odaklanmayı düşünüyorsanız proje yapmak da olmazsa olmaz.