Soru 1 = TERCÜMAN NEDİR?

Tercüman, üst düzey bir programlama dilinde yazılmış kodu yürüten bir bilgisayar programıdır. Çalıştırmadan önce kaynak kodu makine koduna çeviren bir derleyiciden farklı olarak, bir yorumlayıcı

Soru 2 = Python yorumlayıcısının adı nedir?

Python yorumlayıcısının adı "CPython" olarak bilinir. CPython, Python programlama dilinin standart yorumlayıcısıdır ve Python yazılımının orijinal kaynak kodu C dili kullanılarak yazılmıştır. CPython, Python dilinin tüm sürümlerinde kullanılabilir ve açık kaynak kodlu olarak dağıtılmaktadır.

Soru 3 = Python yorumlayıcısı bir tanımlayıcı gördüğünde, hangi kapsamda arar?

Öncelikle tanımlayıcının bulunduğu kapsamda arama yapar. Eğer tanımlayıcı, bu kapsamda bulunamazsa, arama zinciri boyunca daha geniş kapsamlarda arama yapar.

Soru 4 = Bir değişkenin nerede olduğunu veya olup olmadığını görmek için denetlenen çeşitli kapsamlar hangi sıradadır?

tanımlanmış mı?

Python'da, bir değişkenin nerede olduğunu veya olup olmadığını görmek için denetlenen çeşitli kapsamlar, LEGB kuralına göre belirlenir. LEGB, Local (yerel), Enclosing (kapalı), Global (genel) ve Built-in (dahili) kapsamların baş harflerinden oluşan bir kısaltmadır ve değişken isimlerinin arama sırasını belirtir.

Soru 6 = Bir aktivasyon kaydı ne zaman çalışma zamanı yığınına aktarılır?

Bir fonksiyon çağrısı sırasında, çağrıldığı fonksiyonun parametreleri ve yerel değişkenleri için yer ayrılmak üzere bir aktivasyon kaydı oluşturulur. Aktivasyon kaydı, çağrılan fonksiyonun parametreleri, yerel değişkenleri, geri dönüş adresi ve diğer gereksinimleri gibi bilgileri depolar. Aktivasyon kaydı, fonksiyon çalıştırılmaya başladığında çalışma zamanı yığınına eklenir ve fonksiyon çalıştığı sürece yığında kalır.

Soru 7 = Çalışma zamanı yığınından bir aktivasyon kaydı ne zaman çıkarılır?

Çalışma zamanı yığınından bir aktivasyon kaydı (activation record) genellikle fonksiyon çağrısı bittiğinde çıkarılır.

Bir fonksiyon çağrısı sırasında, fonksiyonun çalışması için gerekli olan değişkenler, parametreler ve diğer bilgilerin tutulması için bir aktivasyon kaydı yığına eklenir. Fonksiyon çalışması bittiğinde, aktivasyon kaydı yığından çıkarılır ve programın önceki noktaya geri dönmesini sağlayan geri dönüş adresi kullanılarak işlem devam eder.

Soru 8 = Bir bilgisayarda Heap'te neler bulunur?

1- Dinamik olarak tahsis edilmiş bellek alanları: Programın çalışması sırasında ihtiyaç duyulan ve statik olarak tahsis edilemeyen bellek alanları, Heap'te dinamik olarak tahsis edilir. Bu alanlar, program tarafından kullanıldıkça büyütülebilir veya küçültülebilir.

2-Nesneler: Program tarafından oluşturulan nesneler ve veri yapıları, Heap'te saklanır. Örneğin, bir liste, bir dizi veya bir nesne, Heap'te bellek alanı tahsis edilerek oluşturulur.

3-Bellek yönetim tabloları: Bellek yönetimi sistemi, Heap'teki bellek alanlarının yönetimi için kullanılan veri yapıları ve tablolar da Heap'te saklanır.

4-Boş bellek alanları: Program çalışması sırasında boşalmış ve geri verilmiş bellek alanları da Heap'te saklanır ve yeniden kullanılabilir.

Soru 9 = Çalışma zamanı yığınındaki bir aktivasyon kaydında neler bulunur?

1-Fonksiyonun parametreleri: Fonksiyona verilen parametreler aktivasyon kaydında tutulur. Bu parametreler fonksiyon içinde kullanıldığı için kaydedilmesi gereklidir.

2-Fonksiyonun geçici değişkenleri: Fonksiyonun çalışması sırasında geçici olarak kullanılan değişkenler aktivasyon kaydında saklanır. Bu değişkenler, fonksiyonun çalışması bittikten sonra gereksiz hale gelirler ve kaynakları geri verilir.

3-Geri dönüş adresi: Fonksiyonun çalışması bittiğinde, kontrolün nereye döneceği bilgisi olan geri dönüş adresi aktivasyon kaydında saklanır. Bu bilgi, programın önceki noktaya geri dönmesini sağlar.

Soru 10 = Özyinelemeli bir işlev yazarken, yapmanız gereken iki durum nedir?

kod yaz?

1-Bir durma koşulu belirleyin: Özyinelemeli işlevler, kendilerini tekrar tekrar çağırdıkları için sonsuz bir döngüye neden olabilirler. Bu nedenle, bir durma koşulu belirlemek önemlidir. Durma koşulu, işlevin kendisini tekrar çağırmayı durdurması için bir şarttır. Örneğin, bir sayı sıralama işlevinde, sıralanacak elemanların sayısı 0'a veya 1'e düştüğünde işlev kendisini tekrar çağırmayı durdurabilir.

2-İşlevi kendisiyle çağırın: Özyinelemeli bir işlev, kendisini çağırmak için kendisini kullanır. Bu nedenle, işlevi kendisiyle çağırmalısınız. Bu, işlevin kendisini tekrar tekrar çağırmasına neden olur ve özyinelemeli bir işlevin gereksinimlerini yerine getirir.

Soru 11 = Özyinelemeli bir işlevin temel durumu olmasaydı, bu olduğunda ne olurdu?

isminde?

Özyinelemeli bir işlevin temel durumu (base case) olmaması durumunda, işlev sonsuz bir döngüye girebilir veya çıkış üretemeyebilir. Temel durum, özyinelemeli işlevin en küçük problem boyutuna ulaştığı ve işlevin kendisini tekrar çağırmayı durdurduğu durumdur. Temel durum olmadan, işlev sonsuz bir şekilde kendisini çağırmaya devam edebilir ve bir hata oluşabilir. Örneğin, bir sayının faktöriyelini hesaplayan bir özyinelemeli işlevde, temel durum olarak 0 veya 1 girilen sayılar düşünülebilir. Temel durumun olmaması durumunda, örneğin negatif bir sayı girildiğinde işlev sonsuz bir döngüye girebilir veya hatalı bir çıkış üretebilir

Soru 12 = Özyinelemeli bir işlevin temel durumu olmasaydı, bu olduğunda ne olurdu?

isminde?

Özyinelemeli bir işlevin temel durumu (base case) olmaması durumunda, işlev sonsuz bir döngüye girebilir veya çıkış üretemeyebilir. Temel durum, özyinelemeli işlevin en küçük problem boyutuna ulaştığı ve işlevin kendisini tekrar çağırmayı durdurduğu durumdur. Temel durum olmaması durumunda, işlev sonsuz bir şekilde kendisini çağırmaya devam edebilir ve bir hata oluşabilir. Örneğin, bir sayının faktöriyelini hesaplayan bir özyinelemeli işlevde, temel durum olarak 0 veya 1 girilen sayılar düşünülebilir. Temel durumun olmaması durumunda, örneğin negatif bir sayı girildiğinde işlev sonsuz bir döngüye girebilir veya hatalı bir çıkış üretebilir.

Soru 13 = Type işlevi Python'da ne döndürür? type işlevini bir

programı, Python'un hangi yönünü kullanıyorsunuz?

type() işlevi, Python'un yansıma (reflection) özelliğini kullanarak nesnelerin tip bilgisine erişmek için kullanılır. Python'un bu yönü, programcılara çalışma zamanında nesnelerin özelliklerine ve davranışlarına erişme ve değiştirme olanağı sağlar. Bu özellik, dinamik tip atama ve özyinelemeli işlevler gibi gelişmiş programlama teknikleri için gereklidir.