

Rusya Federasyonu Bilim ve Yüksek Öğrenim Bakanlığının Federal Devlet Bütçe Yüksek Eğitim Kurumu "Ufa Devlet Havacılık Teknik Üniversitesi"

I.V. KIRSANOV

PROFESYONEL İNGİLİZCE
YAZILIM GELİŞTİRLİLERİ İÇİN



Ufa 2022

Rusya Federasyonu Bilim ve Yüksek Öğrenim Bakanlığı Federal Devlet Bütçe Yüksek
Öğretim Kurumu "Ufa Devlet Havacılık Teknik Üniversitesi"

I.V. KIRSANOV

PROFESYONEL İNGİLİZCE
YAZILIM GELİŞİMİ RİCİLERİN İÇİN N

UGATU Yayın ve Yayın Kurulu tarafından lisans ve yüksek lisans
eğitimini yönünde okuyan tam zamanlı ve yarı zamanlı öğrenciler için ders kitabı olarak
onaylanmıştır 09.03.04, 09.04.04 Yazılım mühendisliği

Ağ erişiminin eğitici elektronik sürümü

© DİNLE
ISBN 978-5-4221-1597-6

İnceleyenler:

Devlet Fizik ve Matematik Üniversitesi Matematik ve Bilgi Teknolojileri Fakültesi Dekanı
Profesör Z.Yu.

Kırsanova I.V.

Yazılım geliştiriciler için profesyonel ingilizce: ders kitabı[Elektronik kaynak]/Ufimsk Devlet Havacılık Teknik Üniversitesi-Ufa: UGATU, 2022.-URL:<https://www.ugatu.su/media/uploads/MainSite/Ob%20universitete/>
İzdateli/El_izd/2022-117.pdf

BT uzmanlarının eğitimi yönüne uygun olarak orijinal İngilizce metinleri okuma ve tercüme etme, sözlü ve yazılı formlarda konuşma becerilerini geliştirmek ve profesyonel odaklı yabancı dil iletişim becerilerini geliştirmek amaçlanmaktadır.

Eğitim gören öğrenciler ve lisans öğrencileri için tasarlanmıştır
disiplin "Mesleki faaliyetlerde yabancı dil"
yanı sıra bilgi teknolojisi ile ilgili öğrenciler için.

Elektronik baskının hazırlanmasında aşağı idaki yazılım araçları kullanıldı:

- Adobe Acrobat - metin düzenleyici;
- Microsoft Word bir metin düzenleyicisidir.

Yazar: KirsanovaInnaVyacheslavovna

Düzenleme ve düzen: O.A. Sokolova

Programlama ve bilgisayar tasarımcısı: O.M. Tolkacheva

Tüm hakları Saklıdır. Kitap veya herhangi bir kısmı kopyalanamaz, elektronik veya mekanik olarak, fotokopi, bilgisayar belleğ ine kayıt, çoğaltma veya başka bir şekilde çoğaltılamaz, ayrıca çoğaltılamaz. Yayıncının izni olmadan herhangi bir bilgi sisteminde kullanılamaz. Bir kitabın veya bir bölümünün yayıcının izni olmadan kopyalanması, çoğaltılması ve başka şekillerde kullanılması yasa dışıdır ve cezai, idari ve hukuki sorumluluk gerektirir.

Kullanılarak imzalandı: 30.06.2022

Hacim: 2,7 Mb.

FGBOUVO "Ufa Devlet Havacılık Teknik Üniversitesi" 450008, Ufa, ul. K. Marks, 12. Tel.: +7-908-35-05-007 e-posta: rik@ugatu.su

ÖNSÖZ

'Yazılım Geliştiricileri için Profesyonel İngilizce' ders kitabı, uzmanlık alanı olarak yazılım mühendisliği ini seç en çok öğrenciler için bir kılavuz olarak tasarlanmıştır. İngilizce iletişim becerileri kazanıp ustalaşabilecek ve bunları mesleki alanlarında etkin bir şekilde kullanabileceklerdir. Bu nedenle, ders kitabının temel amacı, öğrencilerin İngilizce dilini çeşitli iletişimsel amaçlar için kullanma becerilerini geliştirmektir.

Kılavuz ayrıca, öğrencilerin İngilizcede uzmanlaşmaları ve geleceğin uzmanlarıyla kültürlerarası iletişim kurmaları için bağimsız çalışmalarını organize etmenize olanak tanır. Üniteenin en başından itibaren öğrenciler, konuya ilgili kelime dağarcığıyla bağimsız olarak çalışma fırsatı verilir.

Kitap, ek okuma için 12 ünite ve metin içerir. Bu metinler, metin anlama ve çeviri becerilerini test etmek ve kontrol etmek için önerilebilir. Her üitede yeterli sayıda sözcük alıştırmaları, tartışmalar için farklı görevler, rapor yazmak ve sunum yapmak için önerilen konular bulunur. Sözcük ve dilbilgisi görevleri 'Yazılım Geliştirme için İngilizce', yabancı dillerin öğrenilmesi ve öğretilmesinin modern ilkelerine göre geliştirilmiştir. Dilbilgisi Revizyonu görevleri tablolar halinde sunulur ve dilbilgisi kalıplarını kullanarak dönüştürme, çeviri ve cümle tamamlama konularında çok çeşitli dilbilgisi görevleri de İngilizceyi pratikte öğrenmek ve kullanmak için çok faydalıdır.

Bu görevlerin, öğrencilerin ve Yüksek Lisans öğrencilerinin çok çeşitli dil becerilerini uygulamalarına izin vermenin iyi bir yolu olabileceğini umuyoruz. Buna karşılık öğrenciler, öğrencilerin öğrenme stratejilerini ve eleştirel düşünme gelişimini destekleyerek otantik dil pratiği i deneyimlerine dahil edebilirler.

Sonuç olarak, mesleki amaçlarla İngilizce öğrenmenizde başarılar dileriz!

ÜNİ TE 1. Yazılım Mühendisliği İnce Giriş

Öğrenme hedefleri

Yazılım mühendisliği ve türleri hakkında temel bilgileri edinmek Tüm yazılımların yazılım mühendisliği gerektirip gerektirmeyi anlamak

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anıtlarını hatırlayın.

Konsept; karmaşık olmak; başvuru; tanım; aletler; güvenlik riskleri; güvenlik açığı; operasyonel yazılım mühendisliği; geçiş yazılım mühendisliği; tekrarlayan; yaşam döngüsü; uygulama; bakım; emeklilik; uyumlu olmak; bağımsız uygulamalar; etkileşimli işlem tabanlı uygulamalar; gömülü kontrol sistemleri; toplu işleme sistemleri

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Yazılım mühendisliği başlı başına bir kavramdır, ancak onu daha iyi anlamak için birlikte nasıl çalışıklarını tam olarak anlayabilmek için terimin her bir bölümünün ne anlaması gerektiğini bilmeniz gereklidir. Basit gibi görünse de anlamak zor olabilir. Bunun nedeni, parçaların sanıldığından daha karmaşık olması ve bir uygulama için yazılım mühendisliği ile çalışmanın zor ve zaman alıcı olmasıdır.

Yazılım mühendisliği inin iki bölümünden biri: yazılım ve mühendislik.

Yazılım, belirli bir işi yapan ve belirli bir gereksinimi karşılayan bir kodlar, belgeler ve tetikleyiciler topluluğuudur.

Mühendislik, en iyi uygulamaları, ilkeleri ve yöntemleri kullanarak ürünlerin geliştirilmesidir.

Yazılım Mühendisliği İ Nedir?

Yazılım ürünlerinin geliştirilmesi ile ilgilenen bir mühendislik dalıdır. Yazılım ve teknoloji değil istikče değil işlen, yıllar boyunca dikkatlice bilenmiş bir dizi ülke, en iyi uygulama ve yöntem dahilinde çalışır.

Yazılım mühendisliği, yaptığı işte güvenilir, verimli ve etkili bir ürüne yol açar. Yazılım mühendisliği,新产品lere yol açabılırken

bunu yapmazsanız, ürün neredeyse her zaman üretim aşamasına geri döner. Peki, yazılım mühendisliği inin tam tanımı nedir?

IEEE tam olarak yazılım mühendisliği yazılım 1. Yazılımın geliştirilmesi, işletilmesi ve bakımına sistematik, disiplinli, ölçülebilir bir yaklaşımın uygulanması; yani, mühendislik in yazılıma uygulanması.

olarak:

Yazılım mühendisliği inin açıktan değil, yazılım mühendisliği i yapılmış her şeyin gerçek makinelerde, içinde değil il, gerçek durumlarda çalışması gerektiğ idir.

Yazılım mühendisliği i, bir uygulamadan bir şirket iç in belirli bir sonuç veya çıktı talebi olduğu unda başlar. BT ekibindeki bir yerden, genellikle CIO'dan, geliştiriciden bir tür yazılım oluşturma iç in bir talep gelir. Yazılım geliştirme ekibi, projeyi gereksinimlere ve adımlara ayırır. Bazen bu çalışma bağ ımsız yüklenicilere, satıcılara ve serbest çalışanlara verilir. Durum böyle olduğu unda, yazılım mühendisliği i araçları, yapılan tüm çalışmaların uyumlu olmasını ve en iyi uygulamaları takip etmesini sağ lamaya yardımcı olur.

Geliştiriciler yazılımlarına ne koyacaklarını nasıl bilirler? Mülakatlar gerçek ekleştirdikten, bilgi topladıktan, mevcut uygulama portföyünü inceledikten ve BT liderleriyle konuştuktan sonra bunu belirli ihtiyaçlara ayırlar. Ardından, yazılımın nasıl oluşturulacağı ina dair bir yol haritası oluşturacaklar. Bu, en önemli kısımlardan biridir, çünkü "işin" çoğu u bu aşamada tamamlanır - bu, aynı zamanda, tipik olarak burada da herhangi bir problemin ortaya çıktığı in anlamına gelir.

Gerçek başlangıç noktası, geliştiricilerin yazılım iç in kod yazmaya başlamasıdır. Bu, çoğu u durumda, kodun mevcut sistemlerle ve bunlarda kullanılan dille uyumlu olması gerektiğ inden, sürecin en uzun kısmıdır. Ne yazık ki, bu sorunlar genellikle projede çok sonrasında kadar fark edilmez ve daha sonra yeniden çalışmanın tamamlanması gereklidir.

Kod, yazıldığ i gibi ve tamamlandıktan sonra, yaşam döngüsünün tüm bölümlerinde test edilmelidir. Yazılım mühendisliği i araçları ile sürekli test ve izleme yapabileceksiniz.

Yazılım Mühendisliği inin

Temelleri Yazılım mühendisliği inin gerçek işi, ürün tasarılanmadan önce başlar ve yazılım mühendisliği inin temelleri, "iş" tamamlandıktan çok sonra da devam ettiğ ini belirtir. Her şey bir ile başlar

yazılımınızın neye ihtiyacı olduğunu dair kapsamlı ve eksiksiz bir anlayış - buna yazılımın yapması gerekenler, çalışması gereken sistem ve iç erdiğin tüm güvenlik dahildir. Güvenlik, yazılım mühendisliğinin temellerinden biridir çünkü geliştirmenin tüm yönleri için çok önemlidir. Kodunuzun nasıl oluşturulduğunu ve herhangi bir güvenlik sorununun nereye düşebileceğini daha iyi anlamana yardımcı olacak araçlar olmadan, ekibiniz geliştirme aşamasında kolayca kaybolabilir.

Yazılım mühendisliği tasarım temelleri, bilgisayar ve sistemler için talimatların oluşturulmasını gerektirir. Bunların çoğu, kapsamlı eğitim almış profesyoneller tarafından kodlama düzeyinde gerçekleştirilecektir. Yine de, yazılım mühendisliğinin her zaman doğrusal bir süreç olmadığıını anlamak önemlidir; bu, tamamlandıktan sonra kapsamlı bir inceleme gerektirdiği anlamına gelir.

Her yazılım yazılım mühendisliği gerektirmez. Tüketiciler tarafından kullanılan basit oyunlar veya programlar, bunlarla ilişkili risklere bağlı olarak mühendislik gerektirmeyebilir. Hemen hemen tüm şirketler, depoladıkları yüksek riskli bilgiler ve oluşturdukları güvenlik riskleri nedeniyle yazılım mühendisliği ine ihtiyaç duyar.

Yazılım mühendisliği, güvenlik açılarını ve riskleri ortaya çıkmadan önce incelemesi gereken özelleştirilmiş, kişiselleştirilmiş yazılımlar oluşturmaya yardımcı olur. Yazılım mühendisliğinin güvenlik ilkeleri gerekli olmadığındade bile, maliyetleri düşürmeye ve müşteri deneyimini iyileştirmeye yardımcı olabilir.

Yazılım Mühendisliği Türleri

Yazılım mühendisliği, bir şemsiye tanım olarak yazılımın tasarımını, geliştirilmesini ve bakımını inceler. Yine de, bir şirketin veya ürünün ihtiyaç duyabileceği farklı yazılım mühendisliği türleri vardır.

Yazılım düşük kaliteli olduğunda veya dağıtımdan önce uygun şekilde incelenmediğinde sorunlar ortaya çıkma eğilimindedir.

Kullanıcı gereksinimlerindeki, tüzüklerdeki ve kullandığı imiz platformlardaki değil işim hızı nedeniyle yazılım mühendislerine çok fazla talep oldu. Yazılım mühendisliği birkaç farklı düzeyde çalışır: Operasyonel Yazılım Mühendisliği: Operasyonel düzeyde yazılım mühendisliği, yazılımın sistemle nasıl etkileşime girdiği ine, bütçede olsun ya da olmasın, kullanılabilirlik, işlevsellik, güvenilirlik ve güvenlik konularına odaklanır. .

Geçiş Yazılım Mühendisliği: Bu tür, yazılımın bir ortamdan diğerine değiştirildiğinde nasıl tepki vereceği odaklanır. tipik olarak

geliştirmede biraz ölçülebilirlik veya esneklik gerektirir.

Yazılım Mühendisliği i Bakımı: Tekrarlayan yazılım mühendisliği i, yazılımın tüm parçaları dē īşikçe mevcut sistem içinde nasıl ç alıştı̄ ina odaklanır.

Yazılım mühendisliği i, analiz, tasarım, geliştirme, test etme, entegrasyon, uygulama, bakım ve hatta kullanımdan kaldırma dahil olmak üzere yazılım geliştirme yaşam dȫngüsünün tüm bölümlerinde işlev görür.

Yazılım mühendisliği inin yeni bir uygulama olmadığı in, ancak sürekli dē īştī in ve düzenli olarak yeni hissedebileceğ ini anlamak önemlidir.

Yazılım evremizdeki her şeye kullanılır, bu nedenle tüm yazılımların düzgün ç alıştı̄ inan emin olmak önemlidir. Aksi takdirde para kaybına, itibar kaybına ve hatta bazı durumlarda can kaybına neden olabilir.

[<https://www.castsoftware.com/glossary/what-is-software-engineering-definition-type-of-basics-introduction>]

1. Metin Tabanlı Ödevler

1.1. Aşağı idaki fiillerden modele göre isimler yapınız ve onları Çevir

Fiil+ -tion (-ation)

Bilgilendirin, yaratın, bağlayın, entegre edin, keşfedin, hazırlayın, belirleyin, gerçekleştirein, ilişkilendirin, uygulayın, çalıştırın.

1.2. Aşağı idaki kelime ve kelimelerin İngilizce karşılıklarını yazınız. kombinasyonlar:

Basit görün; özenli, zaman alıcı; Özel gereksinim; yıllar boyunca honlanmış; tam tanım; güvenlik endişeleri; kapsamlı kontrol; ile ilişkilendirmek Kesim maliyetleri; belli olmak; her şeyi kapsayan bir tanım; ihtiyaç; sistemle etkileşim; işletim yazılımı; geçiş (birinden geçiş

platformlardan dī erine) yazılım geliştirme; Bakım onarım; nice yaklaşım; başlangıç noktası.

1.3. Aşağı idaki kelimeleri tanımlarıyla eşleştirin:

1. Yüksek teknoloji 2.	a) bir bilgisayarda çalıştırılabilen operasyonların aralığı i bilgisayar veya diğ er elektronik sistem
mühendislik 3. işlevsellik 4. dē işim	b) ortaya çıkmak veya daha fazla ö nem kazanmak c) yerine başka bir tane almak veya kullanmak
5. ortaya çıkmak	d) bir bilgisayar iç in bazı programlama dillerinde, genellikle makine dilinde talimatlar e) özellikle elektronikte ileri teknolojik gelişme f) ilgili bir çalışma veya faaliyet alanı
6. kod	belirli bir alanda dē ışıklık veya gelişme g) nicek,
7. uygulama geliştirici 8. seviye	kapsam, derece veya kalite ölçēğ inde bir konum veya aşama h) belirli gereksinimleri karşılamak iç in bilgisayar programları yazan kişi

1.4. Aşağı idaki soruları metne göre cevaplayınız

- 1) Yazılım mühendisliği i ne ile ilgilenir?
- 2) Geliştiriciler iç in başlangıç noktası nedir?
- 3) Tüm yazılımlar yazılım mühendisliği gerektirir mi?
- 4) Yazılım mühendisliği inin önemini nasıl açıklaysınız?
- 5) Yazılım mühendisliği türleri nelerdir?
- 6) Yazılımın tüm böülümlerinde yazılım mühendisliği işlevleri gelişme dē il mi?

1.5. Metni tekrar okuyun ve aşağı idaki ifadelerin doğ ru olup olmadığı ina karar verin veya yanlış.

- 1) Yazılım mühendisliği, yazılım ürünlerinin geliştirilmesi ile ilgilenen bir mühendislik dalıdır ve geliştiriciler iç in en önemli olan güvenlik sorunlarıdır.
- 2) Yazılım mühendisliği, güvenilir, verimli ve yaptığı işte etkilidir.
- 3) Tüm yazılımlar yazılım mühendisliği gerektirir.
- 4) Yazılım yüksek kaliteli olsa ve dağ itimdan önce uygun şekilde incelenmiş olsa bile sorunlar ortaya çıkma ēğ ilimindedir.

- 5) Yazılım mühendisliği i, analiz, tasarım, geliştirme, test dahil olmak üzere yazılım geliştirme yaşam döngüsünün dört bölümünde çalışır.
- 6) Yazılım mühendisliği i yeni bir uygulama değil ildir, ancak sürekli değil işmektedir ve düzenli olarak yeni hissedilebilir.
- 7) Yazılım mühendisliği i her zaman doğrusal bir süreç değil ildir; bu, tamamlandıktan sonra kapsamlı bir inceleme gerektirdiği anlamına gelir.

2. Dilbilgisine Odaklanın

2.1. Parantez içindeki fiillerden birini seçin. Aşağıdaki cümleleri tamamlamak için bunları gerekli forma koyun

- 1) Günümüzde birçok insan (olmak, sahip olmak, yapmak) teknolojiye bağlı ımlıdir, ki onlar için kötü olacak (olacak, olacak, yapacak).
 - 2) iPad'imi yıllardır kullanıyorum (olmak, sahip olmak, yapmak) ve onunla (olmak, sahip olmak, yapmak) çok mutluyum.
 - 3) Dün patronumla konuşma (olma, sahip olma, yapma) şansım olmadığı için, ona Viber'de mesaj atıyorum (olmak, sahip olmak, yapmak).
 - 4) 21. yüzyıl (olmak, sahip olmak, yapmak) en son teknolojilerin çağı.
 - 5) Mobil cihazlar için en son uygulamalar üzerinde araştırma (yapma, yapma, sahip olma) yaparlar.
- 6) Tüm kitaplar çevrimiçi olarak okunabilir (olabilir, yapılabılır, sahip olunabilir).
- 7) Kişisel bilgisayarları kullanmanın herhangi bir dezavantajını görmüyoruz (olmak, sahip olmak, yapmak) hiç.
- 8) Dün bütün gün internette dolaşıyorlar (ol, yap, yap).
- 9) Linda, formda kalmak için her gün (ol, yap, yap) çok fazla egzersiz yap.
- 10) Safari tarayıcısı çevrimiçi öğrecisi (olmak, yapmak, sahip olmak), kullanıcıya onu kullanma konusunda yardım ve destek sağlıyor.

2.2. Pasif yapıları tanımlayın ve cümleleri çevirin

Edilgen sesteki tüm zamanlar yardımcı fiilden oluşur.
karşılık gelen kişi sayısı ve gergin ve anlamsal olmak fiil
geçmiş ortaç / Participle II/ şeklinde fiil.

- 1) Bu yöntem daha önceki bir makalede atıfta bulunulmuştur.
- 2) Bu enstrümana güvenileceğini düşünmüyorum.

- 3) Operasyonel kategorilerde, operasyonlarda yazılım performansını belirleyen faktörler. Özellikle: bütçe, kullanılabilirlik, verimlilik ve doğruluk.
- 4) İş hatasız girilirse, yürütme için seçilecektir.
- 5) Bu elektronik ekipman hızlanmak için tasarlanmıştır.
üretme.
- 6) İş değişkenleri matematiksel fonksiyonlar olarak ifade edilmiş ve edilmekte ve istatistiksel olarak analiz edilmektedir.
- 7) Güvenlikle ilgili yazılım mühendisliği ilkeleri gerekli olmadığıında bile, maliyetleri düşürmeye ve müşteri deneyimini iyileştirmeye yardımcı olabilir.
- 8) Bu röportaj video podcast olarak da kaydedilmiştir. Kontrol et
Software Daily YouTube kanalındaki video.
- 9) Kod yazıldığı gibi ve bir kez yazıldığı gibi test edilmelidir.
tamamlandı - yaşam döngüsünün tüm bölümlerinde.

3. Tartışma

3.1. Tartışma için olası konular

- 1) İş yi bir yazılıminin özellikleri nelerdir? Bu konudaki fikriniz nedir?
- 2) Yazılım mühendisliği ile bilgisayar arasındaki fark nedir?
Bilim?
- 3) Katılıyor musunuz? Neden/neden olmasın?
- 4) Bu mühendislik dalında sizin için ilginç olan nedir?

3.2. Yazılım mühendisliği ile ilgili metnin kısa bir özetini yazın

3.3. Yazılım mühendisleri için en önemli mülakat sorularını kullanarak bir diyalog oluşturun ve dramatize edin

Bir yazılım mühendisinin işe alınması, dikkatle ve üzerinde düşünülerek ele alınması gereken bir süreçtir. İş yi bir yazılım mühendisi şirketinizin büyümeye yardımcı olur, ancak doğrudu becerilere veya iyi bir iş ahlakına sahip olmayan bir yazılım, büyümeyizi yavaşlatabilir ve engellebilir.

Bu nedenle, yazılım mühendislerini başarılı bir şekilde işe almak için işe alım sürecinde sorulacak en iyi soruları bilmelisiniz.

- 1) Yazılım mühendisi olmaya nasıl karar verdiniz?
- 2) Hangi programlama dillerini tercih edersiniz?
- 3) Bir ekip üyesinin kodunu kontrol ederken önemli olan nedir?
- 4) Hangi proje yönetimi araçlarını kullandınız?
- 5) Başarıyla tamamladığınız bir projeden bahsedin.
- 6) Bu işte ne arıyorsunuz?
- 7) Sizi neden işe almalıyız?
- 8) Karşılaştığınız bir sorunu nasıl çözdünüz?
- 9) Şu anda ne üzerinde çalışıyorsunuz?
- 10) Yazılım kalitesini nasıl sağlıyorsunuz?
- 11) Bir ekiple mi yoksa yalnız çalışmaktan mı hoşlanırsınız?
- 12) Kariyer hedefleriniz nelerdir?
- 13) Becerilerinizi nasıl keskin ve güncel tutuyorsunuz?
- 14) Bizim için hangi sorularınız var?

3.4. Uzman Yazılım Mühendisi Nasıl Olunur (ve İstediğiniz Her İşin Alın) konusuna yorumlarınızı yazın.

Gelecekteki Seçim Fikirleri > Nasıl Uzman Bir Yazılım Mühendisi Olunur (ve İstediğiniz Her İşin Alın)



Herkese selam! Birkaç yıl önce Canonical Ltd. için dünyanın en popüler Linux dağıtımını olan Ubuntu'da çalışırken bana sadece bir rüya gibi geliyordu... 2 yıldan fazla bir süredir bu rüyayı yaşıyorum!

Hayalimdeki işi başarmak, başkalarının da hayallerini gerçekleştirmelerine yardımcı olmak için bir kitap yazmam için bana ilham verdi. Kitabın adı: "Nasıl Uzman Yazılım Mühendisi Olunur (ve İstediğiniz Her İşin Alın)". Okuyuculara yeni beceriler edinmenin ve odaklanmış, gerçek ek dünya iş deneyimi kazanmanın bir yolu olarak ücretsiz ve açık kaynaklı yazılım geliştirme dünyasını tanıtarak etkileyici özgeçmişler oluşturmalarına yardımcı olur.

Bunun hakkında ne düşündüğünüzü duymayı gerçekten çok isterim! [11].

4. Ek okuma

4.1. Aşağı İdaki metni okuyun ve çevirin ve başlık I önerin

Yazılım mühendisliği i, yazılım müşterilerinin ve üreticilerinin ihtiyaçlarının yanı sıra pratik maliyet, zamanlama ve güvenilirlik konularını dikkate alan yazılım üretimine sistematik bir yaklaşımındır. Bu sistematik yaklaşımın gerçekekte nasıl uygulandığı i, yazılımı geliştiren kuruluş, yazılımın türüne ve geliştirme sürecinde yer alan kişilere bağlı olarak önemli ölçüde değil işir. Tüm sistemlere ve tüm şirketlere uygun evrensel yazılım mühendisliği yöntem ve teknikleri yoktur. Aksine, son 50 yılda çok çeşitli yazılım mühendisliği yöntemleri ve araçları gelişmiştir. Hangi yazılım mühendisliği yöntem ve tekniklerinin en önemli olduğuunu belirlemeye belki de en önemli faktör, geliştirilmekte olan uygulamanın türüdür.

Aşağı İdakiler dahil birçok farklı uygulama türü vardır: 1.

Bağimsız uygulamalar , PC gibi yerel bir bilgisayarda çalışan uygulama sistemleridir. Gerekli tüm işlevleri içerirler ve bir ağ bağlantılamları gerekmez. Bu tür uygulamalara örnek olarak bir PC'deki ofis uygulamaları, CAD programları, fotoğraf işleme yazılımı vb. verilebilir.

2. Etkileşimli işlem tabanlı uygulamalar , uzak bir bilgisayarda yürütülen ve kullanıcılar tarafından kendi bilgisayarlarından veya terminallerinden erişilen uygulamalardır. Açıktası, bunlar, mal ve hizmet satın almak için uzak bir sistemle etkileşime girebileceğiniz e-ticaret uygulamaları gibi web uygulamalarını içerir. Bu uygulama sınıfı ayrıca, bir işletmenin bir web tarayıcısı veya özel amaçlı istemci programı aracılığıyla sistemlerine ve posta ve fotoğraf raf paylaşımı gibi bulut tabanlı hizmetlere erişim sağladığını iş sistemlerini de içerir. Etkileşimli uygulamalar genellikle her işlemde erişilen ve güncellenen büyük bir veri deposu içerir.

3. Gömülü kontrol sistemleri , donanım cihazlarını kontrol eden ve yöneten yazılım kontrol sistemleridir. Sayısal olarak, muhtemelen diğer sistem türlerinden daha fazla gömülü sistem vardır. Gömülü sistemlere örnek olarak, bir cep telefonu (cep) telefonundaki yazılım, bir arabada kilitlenme önleyici frenlemeyi kontrol eden yazılım ve bir mikrodalga fırında pişirme sürecini kontrol eden yazılım yer almıştır.

4. Toplu işleme sistemleri , verileri büyük gruplar halinde işlemek için tasarlanmış iş sistemleridir. Karşılık gelen çıktıları oluşturmak için çok sayıda bireysel girdiyi işlerler. Toplu sistem örnekleri, telefon faturalandırma sistemleri ve maaş ödeme sistemleri gibi periyodik faturalandırma sistemlerini içerir.

5. Eğ lence sistemleri , öncelikle kişisel kullanım için olan ve kullanıcıyı eğ lendirmeyi amaçlayan sistemlerdir. Bu sistemlerin çoğu u şu veya bu türden oyunlardır. Sunulan kullanıcı etkileşiminin kalitesi, eğ lence sistemlerinin en önemli ayırt edici özelliğ idir.

6. Modelleme ve simülasyon sistemleri, bilim adamları ve mühendisler tarafından birçok, ayrı, etkileşimli nesne içeren fiziksel süreçleri veya durumları modellemek için geliştirilen sistemlerdir. Bunlar genellikle hesaplama açısından yoğundur ve yürütme için yüksek performanslı paralel sistemler gerektirir.

7. Veri toplama sistemleri , bir dizi sensör kullanarak ortamından veri toplayan ve bu verileri işlenmek üzere diğer sistemlere gönderen sistemlerdir. Yazılımın sensörlerle etkileşime girmesi gereklidir ve genellikle bir motorun içi veya uzak bir konum gibi düşmanca bir ortama kurulur. 8. Sistem sistemleri , bir dizi başka yazılım sisteminde oluşan sistemlerdir. Bunlardan bazıları, elektronik tablo programı gibi genel yazılım ürünleri olabilir.

Montajdaki diğer sistemler o ortam için özel olarak yazılmış olabilir.

Tabii ki, bu sistem türleri arasındaki sınırlar bulanık. Bir mobil (cep) telefonu için bir oyun geliştirirseniz, telefon yazılımının geliştiricileriyle aynı kısıtlamaları (güç, donanım etkileşimi) hesaba katmanız gereklidir. Toplu işleme sistemleri genellikle web tabanlı sistemlerle birlikte kullanılır. Örneğin, bir şirkette, seyahat masrafı talepleri bir web uygulaması aracılığıyla sunulabilir, ancak aylık ödeme için bir toplu başvuruda işlenebilir. Yazılım oldukça farklı özelliklere sahip olduğundan, her sistem türü için farklı yazılım mühendisliği teknikleri kullanırsınız. Örneğin, bir otomobildeki gömülü bir kontrol sistemi güvenlik açısından kritiktir ve araca takıldığıında ROM'a yanar. Bu nedenle de ğ iştirmek çok pahalıdır. Böyle bir sistem çok kapsamlı doğrulama ve doğrulama gerektirir, böylece yazılım sorunlarını gidermek için satıştan sonra arabaları geri çağırma şansı en azı indirilir.

Kullanıcı etkileşimi minimum düzeydedir (veya belki de yoktur), bu nedenle kullanıcı arayüzü prototiplemeye dayalı bir geliştirme süreci kullanmaya gerek yoktur.[10]

4.2. Her tür uygulamayı kendi kelimelerinizle açıklayın

ÜNİ TE 2. Yazılım Nedir? Yazılım Türleri

Öğrenme hedefleri

Yazılımın ne olduğunu ve nasıl çalıştığını anlamak

Farklı yazılım türleri hakkında temel bilgiler edinmek

sistem yazılımı ile sistem yazılımı arasındaki farkı anlamak

Uygulama yazılımı

Anahtar kelimeler ve ifadeler. Rusça eşdeğ erlerini verin ve anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Uygulama yazılımı; sistem yazılımı; yürütmek; Kodlar; dil işlemcisi; ara katman yazılımı; montajcılar; derleyiciler; hata ayıklayıcılar; tercümanlar; kaynak kodu; bilgisayarın sabit diski; üst düzey uygulama yazılımı; önyüklemek için; bir cihazda çalıştırılmak için; görüntü düzenleyiciler; nesne kodu; çalıştmak

B metnini okumadan önce videoyu <https://www.https://searchapparchitecture.techtarget.com/definition/software> konuya

ilgili biraz bilgi edinin

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Yazılım, bilgisayarları çalıştmak ve belirli görevleri yürütmek için kullanılan bir dizi talimat, veri veya programdır. Bir bilgisayarın fiziksel özelliklerini tanımlayan donanımın tersidir. Yazılım, bir cihazda çalışan uygulamalara, komut dosyalarına ve programlara atıfta bulunmak için kullanılan genel bir terimdir. Bir bilgisayarın değil işken parçası olarak düşünülebilir, donanım ise değil işmez parça adır.

Yazılımın iki ana kategorisi, uygulama yazılımı ve sistem yazılımıdır. Uygulama yazılımı, bir kullanıcı için veya bazı durumlarda başka bir uygulama için belirli bir işlevi yerine getiren bir bilgisayar yazılım paketidir. Bir uygulama bağımsız olabilir veya uygulamayı kullanıcı için çalıştan bir program grubu olabilir. Modern uygulamalara örnek olarak ofis paketleri, grafik yazılımları, veritabanları ve veri tabanı yönetim programları, web tarayıcıları, kelime işlemciler, yazılım geliştirme araçları, görüntü düzenleyiciler ve iletişim platformları dahildir.

Sistem yazılımı , bir bilgisayarın donanımını ç alıştırmak üzere tasarlanmıştır ve uygulamaların üzerinde çalışması için bir platform sağ lar. Sistem yazılımı, donanım ve yazılımın faaliyetlerini ve işlevlerini koordine eder. Ayrıca, bilgisayar donanımının işlemlerini kontrol eder ve diğ er tüm yazılım türlerinin ç alışabileceğ i bir ortam veya platform sağ lar.

İ şletim sistemi, sistem yazılımının en iyi örneğ idir; diğ er tüm bilgisayar programlarını yö netir. Sistem yazılımının diğ er örnekleri arasında bellenim, bilgisayar dili ç evirmenleri ve sistem yardımcı programları bulunur.

Diğ er yazılım türleri arasında, yazılım geliştiricilerinin ihtiyaç duyduğu u programlama araçlarını sağ layan programlama yazılımı; sistem yazılımı ve uygulamalar arasında yer alan ara katman yazılımı; ve bilgisayar aygıtlarını ve ç evre birimlerini ç alıştıran sürücü yazılımı.

Dil İ şlemci: Bildiğ imiz gibi, sistem yazılımı insan tarafından okunabilen dili bir makine diline dönüştürür ve bunun tersi de geçerlidir. Yani, dönüştürme dil işlemci tarafından yapılır. Java, C, C++, Python, vb. (kaynak kodu olarak bilinir) gibi üst düzey programlama dillerinde yazılmış programları, makineler tarafından kolayca okunabilen (nesne kodu veya makine kodu olarak bilinir) talimat setlerine dönüştürür.

Sürücü yazılımı. Aygit sürücülerini olarak da bilinen bu yazılım, genellikle bir tür sistem yazılımı olarak kabul edilir. Aygit sürücülerini, bir bilgisayara bağı lı aygıtları ve ç evre birimlerini kontrol ederek belirli görevlerini gerçekleştirmelerini sağ lar. Bir bilgisayara bağı lı olan her aygitin ç alışıması için en az bir aygit sürücüsüne ihtiyacı vardır. Önekler, özel oyun denetleyicileri de dahil olmak üzere herhangi bir standart olmayan donanımla birlikte gelen yazılımların yanı sıra USB depolama aygıtları, klavyeler, kulaklıklar ve yazıcılar gibi standart donanımları etkinleştiren yazılımları iç erir.

Ara yazılım. Ara yazılım terimi , uygulama ve sistem yazılımı arasında veya iki farklı türde uygulama yazılımı arasında aracılık yapan yazılımı tanımlar. Örneğ in, ara yazılım, Microsoft Windows'un Excel ve Word ile konuşmasını sağ lar. Aynı zamanda, bir tür işletim sisteme sahip bir bilgisayardaki bir uygulamadan, farklı bir işletim sisteme sahip bir bilgisayardaki bir uygulamaya uzaktan ç alışma isteğ i göndermek için de kullanılır. Ayrıca daha yeni uygulamaların eski uygulamalarla ç alışmasını sağ lar.

Programlama yazılımı. Bilgisayar programcıları kod yazmak için programlama yazılımı kullanır. Programlama yazılımı ve programlama araçları, geliştiricilerin diğ er yazılım programlarını geliştirmesine, yazmasına, test etmesine ve hata ayıklamasına olanak tanır. Programlama yazılımı örnekleri arasında birleştiriciler, derleyiciler, hata ayıklayıcılar ve yorumlayıcılar bulunur.



Şekil 1. İ şte tam yazılım yiğ ininin tam bir resmi

Yazılım nasıl ç alışır?

Uygulama yazılımı, son kullanıcılar için rapor yazma ve web sitelerinde gezinme gibi belirli işlevleri gerçekleştiren birçok programdan oluşur. Uygulamalar, diğer uygulamalar için de görevler gerçekleştirebilir.

Bir bilgisayardaki uygulamalar kendi başlarına çalışmazlar; çalışmak için diğer destekleyici sistem yazılım programlarıyla birlikte bir bilgisayarın işletim sistemine ihtiyaç duyarlar.

Bu masaüstü uygulamaları, bir kullanıcının bilgisayarına yüklenir ve görevleri gerçekleştirmek için bilgisayar belleğ ini kullanır. Bilgisayarın sabit diskinde yer kapıları ve çalışmak için internet bağlantısına ihtiyaç duymazlar.

Ancak masaüstü uygulamaları, üzerinde çalışıkları donanım aygıtlarının gereksinimlerine uymalıdır.

Web uygulamaları ise çalışmak için yalnızca internet erişimi gereklidir; çalıştırmak için donanım ve sistem yazılımına güvenmezler.

Sonuç olarak, kullanıcılar web tarayıcısı olan cihazlardan web uygulamalarını başlatabilir. Uygulama işlevinden sorumlu bileşenler sunucuda olduğundan, kullanıcılar uygulamayı Windows, Mac, Linux veya başka herhangi bir işletim sisteminden başlatabilir.

Sistem yazılımı, bilgisayar donanımı ile uygulama yazılımı arasında yer alır. Kullanıcılar, arka planda çalıştığı ve bilgisayarın temel işlevlerini yerine getirdiği için sistem yazılımıyla doğrudan etkileşime girmez.

Bu yazılım, bir sistemin donanımını ve yazılımını koordine eder, böylece kullanıcılar belirli eylemleri gerçekleştirmek için üst düzey uygulama yazılımını çalıştırabilir. Sistem yazılımı, bir bilgisayar sistemi açıldığında yürütülür ve sistem açık olduğunda sürece çalışmaya devam eder.

Sistem yazılımı ve uygulama yazılımı

Sistem yazılımı	Uygulama yazılımı
Temel sistem kaynaklarını ve süreçlerini yöneten genel amaçlı yazılım	Kullanıcı ihtiyaçlarını karşılamak için belirli görevleri gerçekleştiren sistem yazılımı
Düşük seviyeli montaj dilinde veya makine kodunda yazılmıştır	Python veya JavaScript gibi üst düzey dillerde yazılmış
Belirli donanım ihtiyaçlarını karşılamalıdır; donanım ile yakından etkileşime girer	Donanımı hesaba katmaz ve donanımla doğrudan etkileşime girmez
İşletim sistemi ile aynı anda, genellikle üretici tarafından yüklenir	Kullanıcı veya yönetici gerektiğiinde yazılımı yükler
Bilgisayar her açıldığıında çalışır	Kullanıcı programı tetikler ve durdurur
Arka planda çalışır ve kullanıcılar genellikle buna erişmez	Ön planda çalışır ve kullanıcılar belirli görevleri gerçekleştirmek için doğrudan yazılımla çalışır
Bağimsız çalışır	Çalışırmak için sistem yazılımına ihtiyaç duyar
Sistemin çalışması için gerekli	Sistemin çalışması için gerekli değil

Şekil 2. Sistem ve uygulama yazılımı arasındaki temel farklar

İlk yazılımlar belirli bilgisayarlar için yazılmış ve üzerinde çalıştığı donanımla birlikte satılmıştır. 1980'lerde, yazılımlar disketlerde ve daha sonra CD ve DVD'lerde satılmaya başlandı. Günümüzde çoğu yazılım satın alınmakta ve doğrudan internet üzerinden indirilmektedir. Yazılım, satıcı web sitelerinde veya uygulama hizmeti sağlayıcı web sitelerinde bulunabilir.

[<https://searchapparchitecture.techtarget.com/definition/software>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Veritabanı Yönetimi; değil istirilemez kısım; bağimsız (özerk); gelişmiş metin düzenleyici biçimlendirme; bir platform sağlama; mikroişlemci yazılım; çevre birimleri; komut seti; şeffaf işlem sağlayan ara katman yazılımı

heterojen bir ağ ortamındaki uygulamalar; kulaklıklar; istek gönder; görevleri yerine getirmek; doğrudan sistem yazılımıyla çalışın

1.2. Anlamını göstermek için her kelimeyle bir cümle yazın

- a) sağlamak – sağlayıcı
- b) eşitlendirmek – değil işmez –
varyasyon c) spesifik – spesifikasyon –
belirtmek d) gerektirmek – gereklilik e)
yürütmek – yürütmek – yürütülebilir

1.3. Metinden aşağıda ki kelimeleri anımlarıyla eşleştiriniz.

Kendi kendine yeten, satıcı, makine dili, işlevsellik, uygulama, platform,
gerçekleştirmek için

- a) bir bilgisayarda veya başka bir bilgisayarda çalıştırılabilen işlem aralığı i
elektronik sistem;
- b) başkalarına bağlı olmamak veya onlardan
etkilenmemek; c) belirli bir standartta çalışmak, işlev görmek veya
bir şey yapmak; d) makine kodu; e) bir bilgisayar sisteminin donanımı
için alıstırabileceğiniz yazılım türlerini belirleyen bir standart; f) bir satıcı,
özellikle gayrimenkul; g) belirli bir amacı yerine getirmek üzere tasarlanmış
bir program veya yazılım parçası

amaç.

1.4. Aşağıda ki soruları metne göre cevaplayınız

- 1) Sistem yazılımı ne için tasarlanmıştır?
- 2) Yazılım hangi türleri veya kategorileri içerir?
- 3) Sürücü yazılımının diğer adı nedir?
- 4) Derleyiciler ve derleyiciler programlama örnekleri midir?
yazılım mı, ara katman yazılımı mı?
- 5) Masaüstü uygulamaları hakkında neler söyleyebilirsiniz?
- 6) Yazılımınızın daha ne yapmasını istersiniz?

1.5. Metni tekrar okuyun ve aşağıdaki ifadelerin doğrusu mu yoksa doğrusu mu olduğunu karar verin.
yanlış.

- 1) Yazılım, bir bilgisayarın fiziksel özelliklerini tanımlayan donanımın tersidir ve bir bilgisayarın değil işken parçası olarak düşünülebilir, donanım ise değil işmez kısımdır.
- 2) Bir uygulama hiçbir zaman aşağıdakileri çalıştıran bir program grubu olamaz. kullanıcı içinden uygulama.
- 3) İnternet platformu, modern çağın örneklerinden biridir. uygulamalar.
- 4) Sistem yazılımı, yazılımın faaliyetlerini ve işlevlerini koordine eder.
- 5) C++, düşük seviyeli programlama dillerinden biridir.
- 6) Bilgisayara bağlı her cihaz herhangi bir cihaza ihtiyaç duymaz çalışıması içinden sürücü.
- 7) Uygulama işlevinden sorumlu bileşenler sunucuda olduğundan, kullanıcılar uygulamayı Windows, Mac, Linux veya başka herhangi bir işletim sisteminden başlatabilir.
- 8) 1990'larda disketlerde ve daha sonra CD ve DVD'lerde yazılım satılmaya başlandı.

2. Dilbilgisine Odaklanın

2.1. Aşağıda ki cümleleri tamamlayın ve çevirin (modal fiile dikkat edin)

- 1) Yöneticimiz geçen ay birçok karmaşık pratik sorunu _____ çözüyor (zorunluydu, yapmalı, yapmalı, yapmalı).
- 2) Şimdi bu fenomeni _____ inceliyor (olabilir, olabilir, olabilir).
- 3) Yıllar önce faaliyette olan bir jet motoru (göremez, göremez, görmüş olabilir).
- 4) Kullanıcılar _____ uygulamayı Windows, Mac, Linux veya başka herhangi bir işletim sisteminden başlatır (olmalıdır, yapabilir, yapamaz).
- 5) Masaüstü uygulamaları, üzerinde çalıştıkları donanım aygıtlarının gereksinimlerine uyar (izin verilir, gereklidir, olmamalıdır).
- 6) Yazılım, satıcı web sitelerinde veya uygulama hizmeti sağlayıcı web sitelerinde bulunabilir (can't, must, can, must).

7) Örneğin, İnternet tarayıcı yazılımınız olmadan internette _____ gezinirsiniz ve bu makaleyi okursunuz (yapmamalı, yapamaz, yapamaz).

8) İş büyümesi, yönetilen veri miktarındaki artışla birlikte gelir ve bu da sürekli o _____ büyüyen tüm bilgileri (olabilir, olamaz, olmalıdır) yönetme ihtiyacıyla sonuçlanır.

2.2. Aşağıdaki modal anlamları göstermek için tabloyu inceleyin ve kendi cümlelerinizi oluşturun

a) gerekir, olabilir ve olabilir + mükemmel mastar	olasılık i ifade et veya geçmişle ilgili bir eylemin olasılığı ve genellikle tercüme edilir kelimeler "olmalı Belki".	Kitabını bir yerlerde kaybetmiş olmalı . Kaybetmiş olmalı kitabıınız bir yerde.
b) yapabilir / yapamaz + olamaz + mükemmel mastar	hakkında şüphe ifade etmek yapma imkanı Geçmişteki eylemler ve genellikle tercüme edilir "kelimelerin yardımı olamaz olmak".	Yapmış olamaz bu kadar ciddi bir hata. hayır Belki o izin verildi ciddi hata.
c) gerekir (yapmalı), yapmalı, yapabilir, olabilir + Mükemmel Sonsuz	şunu belirtiyorlar olabilecek eylem ya da sahip olmalı yapılacak, yapılacak.	Devrenin tüm noktalarında akım gücünü değ iştirmiş olmalısınız . Bam değ işmeliydi tüm noktalarda mevcut güç zincirler.

2.3. Aşağıdaki cümleleri Perfect Infinitives ile birlikte modal fiillerle çevirin

1) Sadece köyde göründükleri için yollarını kaybetmiş olmalılar.
gece.

2) Bu ilmi eseri bu kadar kısa sürede tanıdıg i bir adama emanet etmiş olamaz.

3) Yapmış olması gereken her şeyi düşünerek gözlerini onun üzerine diktı.
Onu en son gördüğ ünden beri.

- 4) Onu orada görmüş olamazsun çünkü o yeri iki kere terk etti.
aylar önce.
- 5) Egzersizlerinizde çok fazla hata var. Daha dikkatli olmalıydin.

- 6) Sabah onu odasında bulamadım, bize not bırakmadan gitmiş olmalı.

- 7) Olabilecek bir şeyi gözden kaçırılmış olabilir.
masumiyetini kanıtlaması açısından önemlidir.
- 8) Birkaç kelimeyle onlara ölümcül bir kazanın başına geldiğ ini söyledi.

3. Tartışma

3.1. Yazılım geliştirme ile ilgili soruları cevaplayın

- 1) Bilgisayar yazılımı hakkında ne kadar bilginiz var?
- 2) En sevdiğiniz yazılım parçası nedir?
- 3) Hiç yazılım probleminiz oldu mu?
- 4) En son yazılımları güncel tutuyor musunuz?
- 5) Birçok yazılımın ücretsiz olmasına şaşırdınız mı?
- 6) Hiç korsan yazılım satın aldınız mı? yapar misin?
- 7) Konuşma tanıma yazılımı hakkında ne düşünüyorsunuz?
- 8) Yazılımınızın daha ne yapmasını istersiniz?
- 9) Bill Gates veya Steve Jobs'a hangi soruları sormak istersiniz?
yazılım?

3.2. Tartışma için olası konular

- 1) Bill Gates şunları söyledi: "Üretkenlik yazılımı ve diğer BT araçları hakkında sağlam bir çalışma bilgisi, hemen hemen her kariyerde başarı için temel bir temel haline geldi." onunla aynı fikirde misin?
- 2) Şimdiye kadar yaratılmış en büyük yazılım parçası nedir?
- 3) Microsoft'un neden yazılıma bağlı kaldığını ve hiç gitmediğini düşünüyorsunuz?
Apple gibi bilgisayar üretmeye mi?
- 4) Yazılımın para için iyi bir değer olduğunu düşünüyor musunuz?
- 5) Yazılım tasarılayan insanlar hakkında ne düşünüyorsunuz?

4. Ek okuma

4.1. Aşağıdaki metni okuyun ve çevirin

Tasarım ve Uygulama

Yazılım geliştirme yaşam döngüsü, proje yöneticilerinin yazılım tasarlama ile ilgili aşamaları ve görevleri tanımlamak için kullandığı bir çerçevedir. Tasarım yaşam döngüsündeki ilk adımlar, çabanın planlanması ve ardından yazılımı kullanacak kişilerin ihtiyaçlarının analiz edilmesi ve detaylı gereksinimlerin oluşturulmasıdır. İlk gereksinim analizinden sonra, tasarım aşaması bu kullanıcı gereksinimlerinin nasıl karşılanacağıını belirlemeyi amaçlar.

Bir sonraki adım, geliştirme çalışmasının tamamlandığı ve ardından yazılım testinin gerçekleştiği uygulamadır. Bakım aşaması, sistemi çalışır durumda tutmak için gereken tüm görevleri içerir.

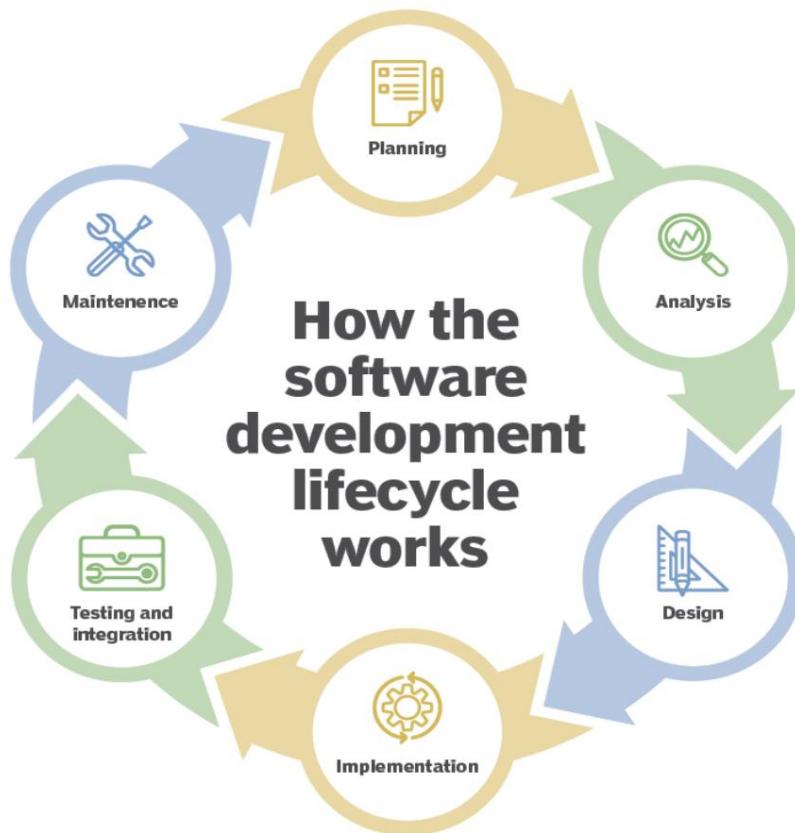
Yazılım tasarımları, uygulanacak yazılımın yapısının, veri modellerinin, sistem bileşenleri arasındaki arayüzlerin ve potansiyel olarak yazılım mühendisinin kullanacağı algoritmaların bir tanımını içerir.

Yazılım tasarım süreci, kullanıcı gereksinimlerini bilgisayar programclarının yazılım kodlamasını ve uygulamasını yapmak için kullanabilecekleri bir forma dönüştürür. Yazılım mühendisleri, yazılım tasarımını yinelemeli olarak geliştirir, ayrıntı ekler ve tasarımını geliştirirken düzeltir.

Farklı yazılım tasarımları türleri şunları içerir: Mimari tasarım. Bu, mimari tasarım araçlarını kullanarak sistemin genel yapısını, ana bileşenlerini ve bunların birbirleriyle olan ilişkilerini tanımlayan temel tasarımdır.

Üst düzey tasarım. Bu, sistemin tüm bileşenleriyle birlikte bir yazılım yığını tarafından desteklenen modüller biçiminde nasıl uygulanabileceğine odaklanan ikinci tasarım katmanıdır. Üst düzey bir tasarım, veri akışı ile sistemin eşitli modülleri ve işlevleri arasındaki ilişkileri tanımlar.

Detaylı tasarım. Bu üçüncü tasarım katmanı, tüm belirtilen mimari için gerekli uygulama ayrıntıları.



Şekil 3. Yazılım geliştirmede yer alan ana adımlar

Yazılım kalitesi nasıl korunur

yazılım kalitesi, yazılımın hem işlevsel hem de işlevsel olmayan gereksinimlerini karşılayıp karşılamadığını ölçer.

Yazılım kalitesi, yazılımın hem işlevsel hem de işlevsel olmayan gereksinimlerini karşılayıp karşılamadığını ölçer. **kalite ölçüm(önlem)** **islevsel gereksinim**

İşlevsel gereksinimler, yazılımın ne yapması gerektiğiini tanımlar. Teknik ayrıntıları, veri işleme ve işlemeyi, hesaplamaları veya bir uygulamanın neyi içermek başarmayı amaçladığını belirten diğer herhangi bir özel işlevi içerir.

bunlar, teknik ayrıntıları, veri manipülasyonunu ve hesaplamaları veya bir tamamlamak uygulamanın neyi baarmayı hedeflediini belirten diğer işlevleri içerir.

Kalite nitelikleri olarak da bilinen işlevsel olmayan gereksinimler, sistemin nasıl çalışması gerektiğiini belirler. İşlevsel olmayan gereksinimler taşınabilirlik, olağanüstü durum kurtarma, güvenlik, gizlilik ve kullanılabilirlik içeriir.

Yazılım testi, yazılım kaynak kodundaki teknik sorunları tespit eder ve çözür ve gereksinimlerini karşıladıktan emin olmak için ürünün genel kullanılabilirliğini, performansını, güvenliğini ve uyumluluğunu değerlendirir.

Yazılım kalitesinin boyutları aşağıdaki idakileri içerir:
özellikler:

Ulaşılabilirlik. Ses tanıma ve ekran büyüteleri gibi uyarlanabilir teknolojilere ihtiyaç duyan bireyler de dahil olmak üzere farklı bir grup insanın yazılımı rahatça kullanabilme derecesi.

Uyumluluk Yazılımın çeşitli alanlarda kullanımına uygunluğu ve farklı işletim sistemleri, cihazlar ve tarayıcılar gibi ortamlar.

Yeterlik. Yazılımın enerji, kaynak, çaba, zaman veya para harcamadan iyi performans gösterme yeteneği.

İşlevsellilik. Yazılımın belirtilen işlevlerini yerine getirme yeteneği.

Yüklenebilirlik. Yazılımın belirli bir ortama kurulabilme yeteneği.

Yerelleştirme. Çeşitli diller, saat dilimleri ve diğer er bir yazılımın içinde çalışabileceğin özellikler.

Sürdürülebilirlik. Özellikler eklemek ve geliştirmek, hataları düzeltmek vb. için yazılımın ne kadar kolay değil istirilebileceği.

Verim. Yazılımın belirli bir yük altında ne kadar hızlı performans gösterdiği. Taşınabilirlik Yazılımın bir yerden başka bir yere kolayca aktarılabilmesi.

Güvenilirlik. Yazılımın gerekli bir işlevi yerine getirme yeteneği ve belirli bir süre için belirli koşullar altında hatasız olarak.

Öçeklenebilirlik. Yazılımın artırma veya artırma yeteneğinin ölçüsü. İşleme taleplerindeki değil ışıklıklere yanıt olarak performansı düşürür.

Güvenlik. Yazılımın yetkisiz kişilere karşı koruma yeteneği ve erişim, gizlilik ve ıhlali, hırsızlık, veri kaybı, kötü amaçlı yazılım vb.

Bir kez dağ itildikten sonra yazılım kalitesini korumak için geliştiriciler, yeni müşteri gereksinimlerini karşılamak ve müşterilerin belirlediği sorunları ele almak için sürekli olarak uyarlamalıdır. Bu, sorunları önlemek için işlevselligini geliştirmeyi, hataları düzeltmeyi ve yazılım kodunu ayarlamayı içerir. Bir ürünün piyasada ne kadar süre dayanacağı, geliştiricilerin bu bakım gereksinimlerine ayak uydurabilme becerisine bağlıdır.

Bakım yapmaya gelince, geliştiricilerin yapabileceği dört tür değil ışıklık vardır: Düzeltici. Kullanıcılar genellikle, kodlama hataları ve yazılımın gereksinimlerini

karşılamasını engelleyen diğer er sorunlar da dahil olmak üzere geliştiricilerin düzeltmesi gereken hataları belirler ve bildirir.

Uyarlanabilir. Geliştiriciler, işletim sisteminin yeni bir sürümünün çıkması gibi değil işen donanım ve yazılım ortamlarıyla uyumlu olduğundan emin olmak için yazılımlarında düzenli olarak değil ışıklık yapmalıdır.

mükemmel. Bunlar, performansı artırmak için kullanıcı arabirimini iyileştirmek veya yazılım kodunu ayarlamak gibi sistem işlevselligi ini artıran değil işikliklerdir.

Önleyici. Bu değil işiklikler, yazılımın başarısız olmasını önlemek için yapılır. ve kodu yeniden yapılandırma ve optimize etme gibi görevleri içerir.

Yazılım lisanslama ve patentler

Yazılım lisansı, kullanımı kısıtlayan yasal olarak bağlayıcı bir belgedir. ve yazılım dağıtımını.

Tipik olarak, yazılım lisansları, kullanıcılarla, telif hakkını ihlal etmeden yazılımın bir veya daha fazla kopyasına sahip olma hakkı sağlar. Lisans, sözleşmeye taraf olan tarafların sorumluluklarını ana hatlarıyla belirtir ve yazılımın nasıl kullanılabileceğine ilişkin kısıtlamalar getirebilir.

Yazılım lisanslama hükmü ve koşulları, genellikle yazılımın adil kullanımını, sorumluluk sınırlamalarını, garantileri, feragatları ve yazılımın veya kullanımının başkalarının fikri mülkiyet haklarını ihlal etmesi durumunda korumaları içerir.

Lisanslar genellikle, onu oluşturan kuruluş, grup veya bireyin mülkiyetinde kalan özel mülk yazılımlar içindir; veya kullanıcıların yazılımı çalıştırabileceğini, inceleyebileceğini, değil istirebileceğini ve dağıtabileceğini ücretsiz yazılım için.

Açık kaynak, yazılımın ortaklaşa geliştirildiği ve kaynak kodunun ücretsiz olarak erişilebilir olduğu bir yazılım türüdür. Açık kaynak yazılım lisansları ile kullanıcılar, özgür yazılıma benzer şekilde yazılımı çalıştırabilir, kopyalayabilir, paylaşabilir ve değil istirebilir.

Son yirmi yılda, yazılım satıcıları yazılım lisanslarını tek seferlik satmaktan bir hizmet olarak yazılım abonelik modeline geçtiler. Yazılım satıcıları yazılımı bulutta barındırır ve abonelik ücreti ödeyen ve yazılıma internet üzerinden erişen müşterilerin kullanımına sunar.

Telif hakkı, başkalarının bir geliştiricinin kodunu kopyalamasını engelleyebilse de, telif hakkı, onların aynı yazılımı kopyalamadan bağımsız olarak geliştirmelerini engelleyemez. Öte yandan bir patent, bir geliştiricinin, yazılımı bağımsız olarak geliştirmiş olsa bile, bir geliştiricinin bir geliştiricinin bir patentte iddia ettiğini yazılımın işlevsel yönlerini kullanmasını engellemesini sağlar.

Genel olarak, yazılım ne kadar teknikse, patent alma olasılığı o kadar yüksektir. Örneğin, bir yazılım ürününe patent verilebilir.

yeni bir tür veritabanı yapısı oluşturur veya bir bilgisayarın genel performansını ve işlevini geliştirir.

[<https://searchapparchitecture.techtarget.com/definition/software>]

4.2. Yazılım geçmiş hakkında bir rapor oluşturun. Yazılım tarihinin bu kısa zaman çizelgesi, raporunuz için hazırlanmanıza yardımcı olacaktır.

Yazılım terimi 1950'lerin sonlarına kadar kullanılmadı. Bu süre zarfında, farklı türde programlama yazılımları oluşturulmuş olsa da, bunlar genellikle ticari olarak mevcut değildi. Sonuç olarak, kullanıcılar - çoğu unlukla bilim adamları ve büyük şirketler - genellikle kendi yazılımlarını yazmak zorunda kaldılar.

21 Haziran 1948. Bir bilgisayar bilimcisi olan Tom Kilburn, İngiltere'deki Manchester Üniversitesi'nde Manchester Baby bilgisayarı için dünyanın ilk yazılımını yazıyor.

1950'lerin başı. General Motors, IBM 701 Elektronik Veri İşleme Makinesi için ilk işletim sistemini oluşturur. General Motors İşletim Sistemi veya GM OS olarak adlandırılır.

1958. İstatistikçi John Tukey, bilgisayar programlama ile ilgili bir makalede yazılım kelimesini kullandı.

1960'ların sonu. Disketler tanıtıldı ve 1980'lerde ve 1990'larda yazılım dağıtmak için kullanıldı.

3 Kasım 1971. AT&T, Unix OS'nin ilk sürümünü yayınladı.

1977. Apple, Apple II'yi piyasaya sürdü ve tüketici yazılımı yola çıktı.

1979. VisiCorp, kişisel bilgisayarlar için ilk elektronik tablo yazılımı olan Apple II için VisiCalc'ı piyasaya sürdü.

1981. Microsoft, ilk IBM bilgisayarlarının çoğu unun üzerinde çalıştığı işletim sistemi olan MS-DOS'u yayınladı. IBM, yazılım satmaya başlar ve ticari yazılımlar ortalama bir tüketiciye sunulur. 1980'ler. Sabit sürücüler PC'lerde standart hale gelir ve üreticiler bilgisayarlarda yazılım paketlemeye başlar.

1983. Özgür yazılım hareketi, Richard Stallman'ın özgürce kopyalanabilen, değil iştirilebilen ve dağıtabilen kaynak kodlu Unix benzeri bir işletim sistemi yaratmaya yönelik GNU (GNU Unix değil) Linux projesiyle başlatıldı.

1984. Mac OS, Apple'in Macintosh serisini alıştırmak için piyasaya sürüldü.

1980'lerin ortası. AutoDes AutoCAD, Microsoft Word ve Microsoft Excel dahil olmak üzere önemli yazılım uygulamaları yayınlandı.

1985. Microsoft Windows 1.0 yayınlandı.

1989. CD-ROM'lar standart hale gelir ve disketten çok daha fazla veri tutar

diskler. Büyük yazılım programları hızlı, kolay ve nispeten ucuz bir şekilde dağıtılabilir.

1991. Açık kaynaklı Linux işletim sisteminin temeli olan Linux çekirdeği yayınlandı.

1997. DVD'ler piyasaya sürüldü ve CD'lerden daha fazla veri tutabiliyor, bu da Microsoft Office Suite gibi program demetlerini tek bir diske koymayı mümkün kılıyor.

1999. Salesforce.com, internet üzerinden yazılım dağıtımına öncülük etmek için bulut bilişimi kullanıyor.

2000. Hizmet olarak yazılım (SaaS) terimi moda oldu.

2007. iPhone piyasaya sürüldü ve mobil uygulamalar yaygınlaşmaya başladı.

2010'dan günümüze. Kullanıcılar internetten ve buluttan yazılım satın alıp indirdikçe DVD'lerin modası geçiyor. Satıcılar abonelik tabanlı modellere geçiyor ve SaaS yaygınlaştı.

ÜNİ TE 3. İ şletim Sistemi

Öğrenme hedefleri

- İşletim sisteminin ne olduğunu ve nasıl çalıştığını anlamak
- İşletim sisteminin farklı özellikleri hakkında temel bilgiler edinmek ve fonksiyonlar ve bilgileri düzenlemek için bir çizelge kullanmak
- İşletim sisteminin rolünü anlamak

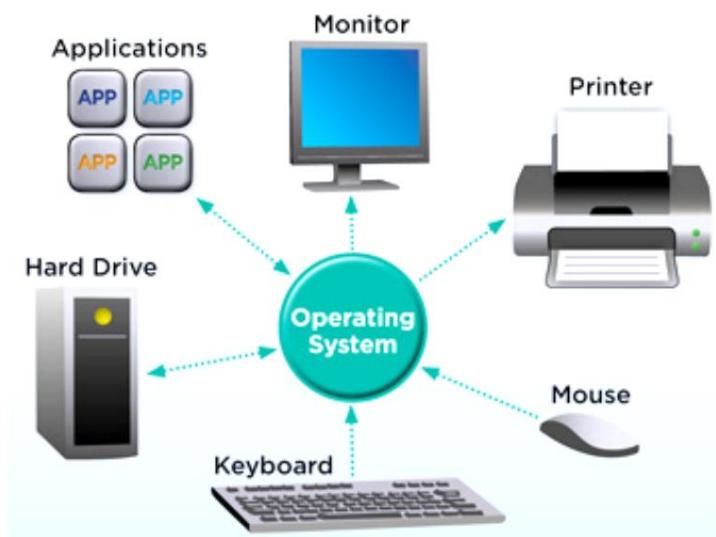
Anahtar kelimeler ve ifadeler. Rusça eşdeğ erlerini verin ve anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Çekirdek yazılım programı; koşmak; tutarlı bir ortam; etkileşim; komut satırı arayüzü; grafiksel kullanıcı arayüzü; yararlanmak; koordine olmak; aygit sürücüler; dosyalar ve Klasörler; yeniden adlandırmak için; sistem sağ ligini izlemek için; verileri almak için; yazılım uyumluluğu; açık kaynak; sahibi olmak para almak için; uyumsuz; baskı; rasgele erişim belleği.

İşletim sistemleri hakkında bilgi almak için <https://edu.gcfglobal.org/en/computerbasics/understanding-operating-systems/1/> adresindeki videoyu okumadan önce. Rusçaya çevir

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

İşletim Sistemi (OS), donanım üzerinde çalışan ve kullanıcının donanımla etkileşime girerek komut gönderebilmesi (giriş) ve sonuç alabilmesi (çıktı) yapabilmesini sağlayan temel yazılım programlarından biridir. Diğer yazılımların komutları yürütmesi için tutarlı bir ortam sağlar. Dolayısıyla işletim sisteminin sistem donanımının, diğer yazılımların ve kullanıcının iletişim kurduğu merkezde hareket ettiğini söyleyebiliriz. Aşağıdaki şekil, işletim sisteminin temel çalışmasını ve farklı donanım veya kaynakları nasıl kullandığını gösterir.



Şekil 4. Çekirdek parça olarak çalışan işletim sistemi

İşletim Sisteminin Temel İşlevleri

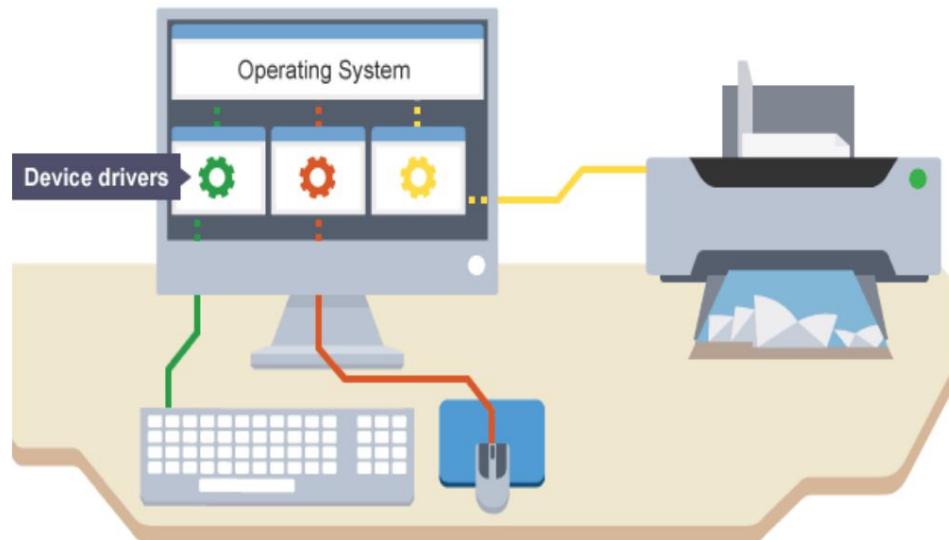
Herhangi bir işletim sisteminin temel beş temel işlevi aşağıda listelenmiştir:

1. Kullanıcı ve donanım arasındaki arabirim: Bir işletim sistemi, kullanıcı ve makine arasında bir arabirim sağlar. Bu arayüz, kullanıcıların işletim sistemiyle etkileşim kurmak için ekran öğelerini tıklattığı bir grafik kullanıcı arayüzü (GUI) veya kullanıcıların işletim sistemine şunu söylemek için komut satırı arayüzüne (CLI) komutlar yazdığını bir komut satırı arayüzü (CLI) olabilir. birleşmeler yap.



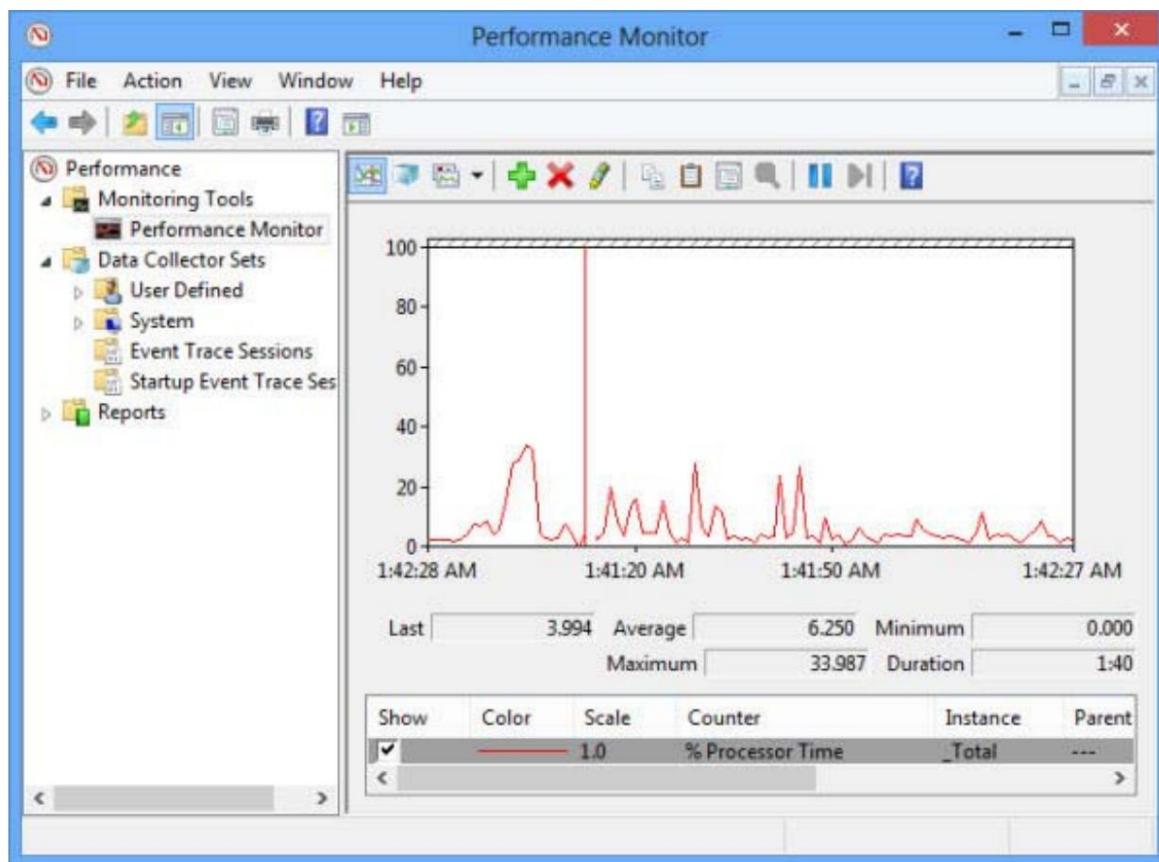
Şekil 5. GUI ve CLI

2. Koordinat donanım bileşenleri: Bir işletim sistemi, aşağıda kılınan koordinasyonunu sağlar :
donanım bileşenleri. Her donanım aygıtı farklı konuşuyor
dil, ancak işletim sistemi onlarla konuşabilir
aygit sürücülerini adı verilen özel çeviri yazılımı. Her donanım
Bileşen, İşletim sistemleri için farklı sürücülere sahiptir. Bu sürücüler
diğer yazılımlar arasındaki iletişimini başarılı kılmak ve
donanım.



Şekil 6. İşletim Sistemi ve Donanım aygıtları arasındaki Aygit Sürücülerı

3. Yazılımın çalışması için ortam sağlayın: Bir işletim sistemi, yazılım uygulamalarının çalışması için ortam. Bir uygulama yazılım, belirli bir görevi gerçekleştirmek için kullanılan belirli bir yazılımdır. Windows ve macOS gibi GUI işletim sistemlerinde, uygulamalar tutarlı, grafiksel bir masaüstü ortamında çalıştırın.
4. Veri yönetimi için yapı sağlayın: Bir işletim sistemi görüntülenir veri yönetimi için yapı/dizinler. Dosyayı görebiliriz ve klasör listeleri ve bu dosya ve klasörler üzerinde işlem yapın (taşı, kopyalayın, yeniden adlandırın, silin ve diğerleri).
5. Sistem sağlığını ve işlevselliliğini izleyin: İşletim sistemi sağlığını izler bize ne kadar iyi (ya da değil) performans gösterdiği dair bir fikir verir. CPU'muzun ne kadar meşgul olduğunu veya sabit disklerimizin ne kadar hızlı veri aldığıını veya ağ kartımızın ne kadar veri gönderdiğiini vb. görebilir ve ayrıca kötü amaçlı yazılımlara karşı sistem etkinliğini izler.



Şekil 7. Windows'ta Performans İzleyicisi

İşletim Sistemi Özellikleri

İşletim sistemleri, lisanslama, yazılım uyumluluğu ve karmaşıklık olan üç temel özelliği göre farklılık gösterir.

Lisanslama

Temel olarak üç tür işletim sistemi vardır. Biri Açık

Kaynak İşletim Sistemi, bir diğer Ücretsiz İşletim Sistemi ve üçüncü Ticari İşletim Sistemidir.

Linux, açık kaynak kodlu bir işletim sistemidir; bu, herkesin onu indirebileceği ve de facto istirebileceği anlamına gelir, örneğin Ubuntu vb.

Ücretsiz bir işletim sisteminin açık kaynak olması gerekmek ve kullanmak ücretsizdir ancak de facto istiremezler. Örneğin, Google, Chrome OS'nin sahibidir ve kullanımı ücretsiz hale getirir.

Ticari işletim sistemleri, onlar için para alan şirktlere aittir. Önekler arasında Microsoft Windows ve Apple macOS bulunur. Bunlar, İşletim sistemlerini kullanma hakkı (veya lisansı) için ödeme yapmayı gerektirir.

Yazılım Uyumluluğu

Geliştiriciler, aynı işletim sistemi türü içinde farklı sürümlerde uyumlu veya uyumsuz olabilecek yazılımı yapar.

ancak diğ er işletim sistemi türleriyle uyumlu olamazlar. Her işletim sistemi türünün kendi yazılım uyumluluğu vardır.

Karmaşıklık

İşletim sistemleri temel olarak biri 32-bit ve diğ eri 64-bit olmak üzere iki sürüm halinde gelir. Bir işletim sisteminin 64-bit sürümü en iyi şekilde rasgele erişim belleğ ini (RAM) kullanır. 64 bit CPU'lu bir bilgisayar 32 bit veya 64 bit işletim sistemi çalıştırabilir, ancak 32 bit CPU'lu bir bilgisayar yalnızca 32 bit işletim sistemi çalıştırabilir.



Şekil 8. 32-bit ve 64-bit Windows işletim sistemi

[<https://medium.com/computing-technology-with-it-fundamentals/operating-system-its-functions-and-characteristics-c0946e4215c6>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Talimatları gönderin; bir ortam sağlamak; öğelere tıklayın ekran; çeviri yazılımı; yerine getirmek belirli bir görev; Veri yönetimi; dosyaları sil; hareket; HDD; yazdırma komutları; kötü amaçlı yazılım; temel özellikleri; karmaşıklık; ücretsiz ameliyathane sistem; kullanım hakkı; uyumluluk; kullanmak en iyi yol.

1.2. Metinden aşağı idaki kelimeleri anlamlarıyla eşleştiriniz.

1. Aygit sürücüsü a)	b) bir bilgisayar tarafından kullanılan en küçük bilgi parçası. bilgisayar.
2. Bit	b) Donanımı, yazılımı ve kaynakları yö neten ve diğ er yazılımlar iç in hizmetler sağ layan sistem yazılımı. Ayrıca genellikle son kullanıcı iç in bir arayüz sağ lar.
3. Komut satırı arayüzü	c) kullanıcının yapmasını istemediğ i, yapmasını istemediğ i ve çoğ u zaman çok geç olana kadar bilgisinin olmadığı i bir şeyi yapmak iç in tasarılanmış herhangi bir yazılım. d) Zaman, adım
4. GUI	sayıları veya aritmetik işlemler veya gereken bellek alanı ile ölçülen matematiksel olarak ortaya konan problemleri çözmedeki zorluk seviyesi. e) kullanıcının bir bilgisayar sistemine giriş yapmadan önce tıklayıp simgelere tıklamayı veya bir menüden seçenekleri seç meyi iç erir.
5. Kötü amaçlı yazılım	
6. Karmaşıklık f)	bir program ile kullanıcıyı arasında yalnızca metinsel girdi ve çıktı olmak üzere bir iletişim aracı. Komutlar b temelli ve yay azen yla gittir ve program tarafından yorumlanır ve yürütülür. Sonuçlar metin veya grafik olarak terminale gönderilir. g) Bu, genellikle, ekirdeğ oluşturulduğ unda bağ lantılı oldukları işletim sistemi ekirdeğ inin en düşük seviyesinin bir parçasını oluşturur. Daha
7. İ şletim sistemi	yeni olanlardan bazıları, dosyalardan yüklenebilen yüklenebilir aygit sürücülerini sistemlerine sahiptir.

1.3. Aşağı idaki soruları metne göre cevaplayınız

- 1) İ şletim sistemleri ile bilgisayar arasındaki ilişki nedir?
donanım?
- 2) Bir işletim sisteminin temel amacı nedir?
- 3) Bir işletim sisteminin kaç temel işlevi vardır?
- 4) Hangi cihazlar birbirleri arasındaki iletişimini başarılı kılar?
yazılım ve donanım?

- 5) Bir işletim sistemi ne görüntüler?
- 6) Açık Kaynak İ şletim Sistemi ile Ücretsiz İ şletim Sistemi arasında herhangi bir fark var mı?
- 7) Aygıt sürücüsü yazılımı ne işe yarar?

1.4. Metni tekrar okuyun ve aşağı idaki ifadelerin doğ ru mu yoksa doğ ru mu olduğunu karar verin.
yanlış.

- 1) Bir işletim sistemi, diğ er yazılımların komutları yürütmesi için tutarlı bir ortam sağ lar.
- 2) Temel olarak üç çeşit İ şletim sistemi vardır. Biri Açık Kaynak İ şletim Sistemi, diğ eri Ücretsiz İ şletim Sistemi ve üçüncü Ağ İ şletim Sistemi.
- 3) Bir işletim sisteminin 64-bit sürümü, salt okunur belleğ i (ROM) en iyi şekilde kullanır.
- 4) Her işletim sistemi türünün kendi yazılım uyumluluğ u vardır.
- 5) Windows ve macOS gibi GUI işletim sistemlerinde uygulamalar grafiksel bir masaüstü ortamında cağırlılamaz.
- 6) Her donanım bileşeninin İ şletim için farklı sürücülerini vardır.
- 7) Linux, ticari işletim sistemlerinin örneklerinden biridir.
- 8) OS, sistemimizin donanımının sağ lığ inizi izler.

2. Dilbilgisine Odaklanın

2.1. Aşağı idaki cümlelerde yüklemlerin zaman ve seslerini tanımlayıp Rusçaya çevirin.

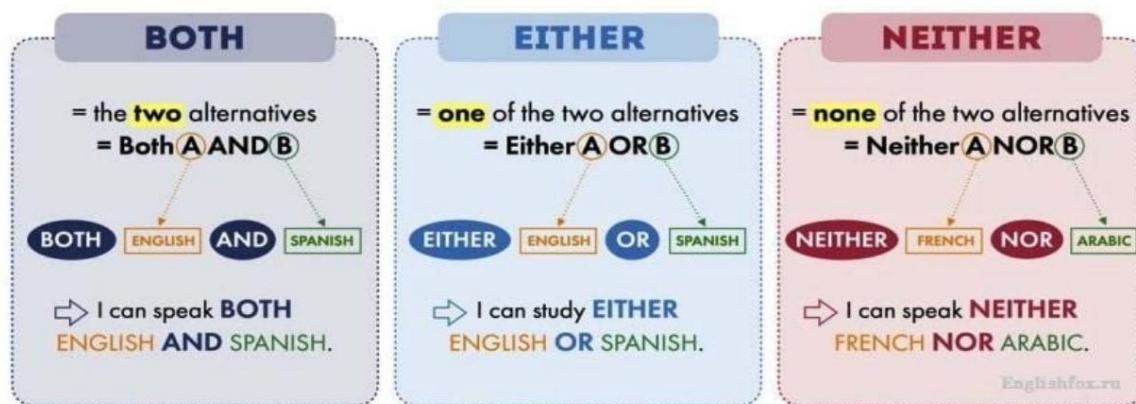
- 1) Aslında, "bilgisayar" teriminin yerini daha hızlı uygun terim "elektronik veri işleme makinesi".
- 2) Son zamanlarda, manyetik disk dosyası hafızalarını kullanabilecek bazı ikili makineler duyuruldu.
- 3) Bu tür bilgisayarlar, son derece yüksek kapasiteli ve erişim hızına sahip bir disk dosyası ile donatılacaktır.
- 4) Son birkaç yılda birkaç tasarım geliştirildi ve bazıları aslında inşa edilmiştir.
- 5) İ ş deð işkenleri matematiksel fonksiyonlar olarak ifade edilmiş ve edilmekte ve istatistiksel olarak analiz edilmektedir.
- 6) Sekiz seçenek konuþmacıdan aşağı idakileri dikkate almaları istenmiþtir:

Yönetimle ilgili faaliyetlerde bilgisayarın potansiyelleri ve sınırlamaları.

- 7) Mekanik olmayan bir yazıcı tasarlama sorunu zaten merkezi araştırma laboratuvarında okudu.
- 8) Bilgisayara cismin konum, veri ve hız vektörleri verilir.
Belirli bir süre için uydu

2.2. İkisinden birini veya ikisini birden ekleyin. Cümleleri Rusçaya Çevir

BOTH – EITHER – NEITHER



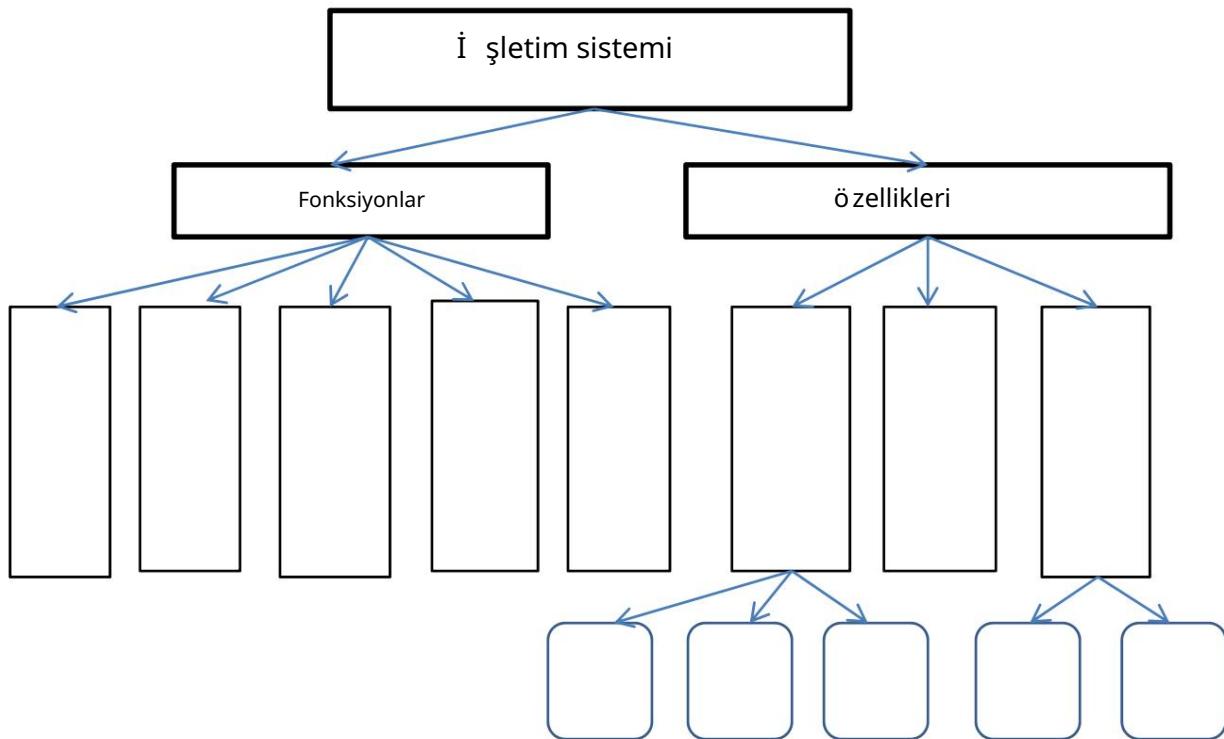
- 1) ...işletim sistemleri (Linux ve UBUNTU) çok iyi.
- 2) Windows veya macOS kurmak istemiyorum.
- 3) Onlar... güldüler ve programcılardan biri başını eğdi.
- 4) Biz ... ofisteydik, ama ... bir süre konuştuğum.
- 5) ...özür dile, yoksa seninle bir daha asla konuşmam.
- 6) Bir sunum hazırlayabilir misiniz?
- 7) ...öğrenciler testi geçti.
- 8) Her ne düşünüyorsan ... yanlışlıyorsun.
- 9) ... o... Projemizin yöneticisiyle konuşmaya gittim.
- 10) Rapordan ve sunumdan keyif alıyorum.

3. Tartışma

3.1. Aşağı idaki soruları grup arkadaşlarınızla tartışın

- 1) Çalışması olmasaydı bir bilgisayarı kullanmak nasıl farklı olurdu? sistem? Programlama nasıl farklı olurdu?
- 2) İ şletim sistemi olmasaydı bir bilgisayarı kullanmak nasıl farklı olurdu? sistem? Programlama nasıl farklı olurdu?
- 3) İnsanların bilgisayarlardan daha iyi yaptığı birkaç zihinsel görevi listeleyin. Liste bilgisayarların insanlardan daha iyi yaptığı bazı zihinsel görevler. Yapabilir misin iki öğedeki öğeleri ayırt eden genel özelliklerini bulun listeler?
- 4) Bazı popüler İ şletim Sistemleri arasında Linux İ şletim Sistemi, Windows İ şletim Sistemi, VMS, OS/400, AIX, z/OS, vb. Nedir? Ev bilgisayarınızda işletim sistemi var mı? Ondan memnun musun? Sahip olmak herhangi bir sıkıntı yaşadın mı?

3.2. Metindeki bilgilere göre grafiği tamamlayın ve işletim sistemlerinin işlevleri ve özellikleri hakkında konuşun



4. Ek okuma

4.1. Bazı İşletim Sistemleri türleri hakkında aşağıdaaki metni okuyun ve çevirin

Toplu işletim sistemi

Toplu işletim sisteminin kullanıcıları bilgisayarla doğrudan etkileşime girmez. Her kullanıcı, delikli kart gibi evrim dışı bir cihazda işini hazırlar ve bilgisayar operatörüne teslim eder. İşlemeyi hızlandırmak için benzer ihtiyaçlara sahip işler bir araya toplanır ve bir grup olarak hazırlanır. Programcılar programlarını operatöre bırakır ve operatör benzer gereksinimleri olan programları grulara ayırır.

Zaman paylaşımı işletim sistemleri

Zaman paylaşımı, çeşitli terminallerde bulunan birçok kişinin aynı anda belirli bir bilgisayar sistemini kullanmasını sağlayan bir tekniktir.

Zaman paylaşımı veya çoklu görev, çoklu programmanın mantıksal bir uzantısıdır. İşlemcinin aynı anda birden fazla kullanıcı arasında paylaşılan zamanına zaman paylaşımı denir.

Çok Programlı Toplu Sistemler ve Zaman Paylaşımı Sistemler arasındaki temel fark, Çok Programlı toplu sistemlerde amaç işlemci kullanımını en üst düzeye çıkarmak iken, Zaman Paylaşımı Sistemlerde amaç yanıt süresini en aza indirmektir.

CPU tarafından aralarında geçiş yapılarak birden fazla iş yürütülür, ancak geçişler çok sık gerçekleşir. Böylece kullanıcı anında yanıt alabilir. Örneğin, bir işlem işlemede, işlemci her kullanıcı programını kısa bir patlama veya hesaplama kuantumunda yürütür. Yani, n kullanıcı varsa, her kullanıcı bir zaman kuantumu alabilir. Kullanıcı komutu gönderdiğinde, yanıt süresi en fazla birkaç saniyedir.

İşletim sistemi, her kullanıcıya zamanın küçük bir bölümünü sağlamak için CPU zamanlaması ve çoklu programlama kullanır. Öncelikle toplu sistemler olarak tasarlanan bilgisayar sistemleri, zaman paylaşımı sistemlere dönüştürülmüştür.

Dağ İtilmiş İşletim Sistemi

Dağ İtilmiş sistemler, birden çok gerçek zamanlı uygulamaya ve birden çok kullanıcıya hizmet için birden çok merkezi işlemci kullanır. Veri işleme işleri, işlemciler arasında buna göre dağıtıılır.

İşlemciler, eşitlik iletişim hatları (yüksek hızlı veri yolları veya telefon hatları gibi) aracılığıyla birbirleriyle iletişim kurar. Bunlar, gevşek bağlı sistemler veya dağıtılmış sistemler olarak adlandırılır. Dağ İtilmiş bir sistemdeki işlemcilerin boyutu ve işlevi değil işebilir. Bu işlemciler siteler, düğümler, bilgisayarlar vb. olarak adlandırılır.

Ağ İşletim Sistemi Ağ

İşletim Sistemi bir sunucu üzerinde çalışır ve sunucuya verileri, kullanıcıları, grupları, güvenlik i, uygulamaları ve diğer ağ işlevlerini yönetme yeteneği sağlar. Ağ işletim sisteminin birincil amacı, bir ağ daki, tipik olarak yerel alan ağ (LAN), özel ağ veya diğer ağlardaki birden çok bilgisayar arasında paylaşılan dosya ve yazıcı erişimine izin vermektir.

Ağ işletim sistemlerine örnek olarak Microsoft Windows Server 2003, Microsoft Windows Server 2008, UNIX, Linux, Mac OS X, Novell NetWare ve BSD kullanılabilir.

Gerçek Zamanlı İşletim Sistemi

Gerçek zamanlı sistem, girdileri işlemek ve bunlara yanıt vermek için gereken zaman aralığıının çevreyi kontrol edecek kadar küçük olduğunu bir veri işleme sistemi olarak tanımlanır. Sistemin bir girdiye yanıt vermesi ve gerekli güncellenmiş bilgilerin görüntülenmesi için geçen süre yanıt süresi olarak adlandırılır. Dolayısıyla bu yöntemde yanıt süresi, çevrimiçi işlemeye kıyasla çok daha kısalıdır.

Gerçek zamanlı sistemler, bir işlemcinin çalışması veya veri akışı konusunda katı zaman gereksinimleri olduğunu unda kullanılır ve gerçek zamanlı sistemler, özel bir uygulamada bir kontrol cihazı olarak kullanılabilir. Gerçek zamanlı bir işletim sistemi, iyi tanımlanmış, sabit zaman kısıtlamalarına sahip olmalıdır, aksi takdirde sistem başarısız olur. Örneğin Bilimsel deneyler, tıbbi görüntüleme sistemleri, endüstriyel kontrol sistemleri, silah sistemleri, robotlar, hava trafik kontrol sistemleri vb.

İki tür gerçek zamanlı işletim sistemi vardır.

Zor gerçek zamanlı sistemler, kritik görevlerin zamanında tamamlanmasını garanti eder. Zor gerçek zamanlı sistemlerde ikincil depolama sınırlıdır veya eksiktir

ve veriler ROM'da saklanır. Bu sistemlerde sanal bellek neredeyse hiç bulunmaz.

Yumuşak gerçek zamanlı sistemler daha az kısıtlayıcıdır. Kritik bir gerçek zamanlı görev, diğ er görevlere göre öncelik kazanır ve tamamlanana kadar önceliği korur. Yumuşak gerçek zamanlı sistemler, katı gerçek zamanlı sistemlere göre sınırlı faydaya sahiptir. Örneğ in, multimedya, sanal gerçek eklik, denizaltı keşifleri ve gezegen gezicileri gibi Gelişmiş Bilimsel Projeler vb. [https://www.tutorialspoint.com/operating_system/os_types.htm]

4.2. İşletim sistemi türlerinin avantajları ve dezavantajları hakkında bilgi içeren tabloyu doldurun. Gerekirse bazı İ nternet kaynaklarını kullanabilirsiniz.

Türler	Avantajlar	Dezavantajları
Toplu işletim sistemi...		Kullanıcı ve iş arasındaki etkileşim eksikliği. ...
Dağıtılmış işletim sistem	için daha iyi hizmet müşteriler.
Zaman paylaşımı işletim sistemleri	Hızlı yanıt avantajı sağlar. ...	Güvenilirlik sorunu. ...
Ağ işletimi sistem	Güvenlik sunucusu sunucu yönetilir. ...	Bir sunucu satın alma ve çalıştırmanın yüksek maliyeti. ...

4.3. Aşağıdaki soruları cevaplayın ve cevabınızı açıklayamaya çalışın

Kendini test et

Soru 1. Aşağıdakilerden hangisi yazılım değildir?

- (A) MS-Word
- (B) MS-Excel (C)
- Klavye (D)

Microsoft Windows

Çözümü: _____

Soru 2. Aşağıdakilerden hangisi kullanıcı ile bilgisayar donanımı arasında bir arayüz görevi görür?

- (A) Monitör
- (B) İşletim sistemi (C)
- Kullanıcı iş parçası (D) Uygulama programı

Çözüm: _____

Soru 3. Bilgisayarın işleyebildiği veya alıştırılabildiği tek dil denir.
_____ ?

- (A) Makine dili
- (B) Normal dil
- (C) Bilgisayar dili
- (D) Üst düzey dil

Çözüm: _____

Soru 4. Bir bilgisayarın işlemlerini kontrol etmek için aşağıdakilerden hangisi kullanılır?

- (A) Uygulama Yazılımı (B) Sistem Yazılımı (C) Yardımcı Yazılım (D) Dil İşlemci

Çözümü: _____

Soru 5. Aşağıdakilerden hangisi belirli bir sorunu çözmek veya belirli bir görevi yapmak için tasarlanmıştır?

- (A) Dil İşlemcisi (B) Uygulama Yazılımı (C) Sistem Yazılımı (D) Yardımcı Yazılım

Çözümü: _____

Soru 6. Aşağıdakilerden hangisi işletim sistemine örnek değildir?

- (A) Linux
- (B) Apple macOS
- (C) Microsoft Windows,
- (D) Yukarıdaki Çözümlerin hiçbiri:

Soru 7. Aşağı idakilerden hangisi bir dil işlemcisidir?

- (A) C++ programlama dili (B)
Derleyici (C) Linux (D) Yukarıdaki
Çözümlerin tümü:

[<https://www.geeksforgeeks.org/software-and-its-types/>]

4.4. Aşağı idaki cümleleri çevirin

1. İ şletim sistemi ana yazılımdır
tüm donanımı ve diğ erlerini yöneten yazılım
bir bilgisayarda yazılım.
2. "OS" olarak da bilinen işletim sistemi, bilgisayarın donanımıyla etkileşime girer ve
kullanılabilecek hizmetler sağ lar
uygulamalar.
3. İ şletim sistemleri ayrıca birçok
ortak sistem hizmetleri, kitaplıklar ve uygulama programlama arabirimleri gibi yazılım
ürünleri
(API) geliştiricilerin yazmak için kullanabileceğ i
işletim sisteminde çalışan programlar. 4. Çok u yazılım uygulaması,
işletim sistemine izin veren işletim sistemleri
çok iş yap.
5. Örneğ in, Minecraft'ı başlattığ ınızda,
işletim sistemi.
6. Minecraft herkesin nasıl ç alıştığ ini tam olarak bilmek zorunda değ il
ayrı donanım bileşeni.
7. Minecraft, çeşitli işletim özelliklerini kullanır
sistem ve işletim sistemi bunları çevirir
düşük seviyeli donanım talimatları. 8. Aynı anda çalışan
programların sayısına göre
işletim sistemleri tek görevli ve
çoklu görev.

Testin anahtar cevapları 4.3.

1. Doğru seçenek C'dir, yani Klavye. Çünkü klavye bir donanım aygıtı (giriş aygıtı) olduğu için yazılım değildir.
2. Doğru seçenek B'dir, yani İşletim Sistemi. Çünkü bir işletim sistemi, kullanıcıya bilgisayar sistemi ile etkileşime girmesine yardımcı olan bir arayüz sağlar.
3. Doğru seçenek A'dır, yani Makine dili. Bilgisayarın işleyebileceği veya çalıştırabileceği tek dile makine dili denir, çünkü bu dil bilgisayara açıkça ne yapacağıını söylemeye yeteneği sahiptir.
4. Doğru seçenek B'dir, yani Sistem Yazılımı. İki tür yazılım vardır: sistem yazılımı ve uygulama yazılımı. Sistem Yazılımı, işlemleri kontrol etmek için kullanılır ve ayrıca bir bilgisayarın dahili işleyişini ve donanım aygıtlarını kontrol eder.
5. Doğru seçenek B'dir, yani Uygulama Yazılımı. Çünkü özel işlevleri yerine getiren veya bilgisayarın temel işleyişinden çok daha fazla işlev sağlayan yazılımlar uygulama yazılımlarıdır.
6. Doğru seçenek D'dir, yani, yukarıdakilerin hiçbiri. Çünkü Linux, Apple macOS, Microsoft Windows işletim sistemlerine örnektir.
7. Doğru seçenek B'dir, yani Derleyici. Çünkü bir dil işlemcisi, program kodunu makine koduna dönüştürmek için tasarlanmış veya kullanılmıştır. Yani derleyici bir dil işlemcisidir ve C/C++ programlama dilinde kullanılır.

ÜNİ TE 4. Bilgisayar Programlama Nedir? Kodlama ve Programlama

Öğrenme hedefleri

- Bilgisayar programlama hakkında temel bilgileri edinmek
- Programlamanın temel amacını anlamak Kodlama ve programlama arasındaki farkı anlamak

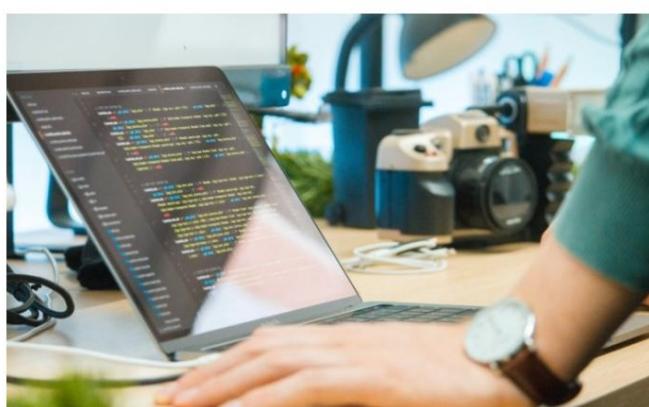
Anahtar kelimeler ve ifadeler. Rusça eşdeğerlerini verin ve anahtar kelimelerin ve ifadelerin anımlarını hatırlayın.

Programlama dili; başarmak; anlaşılır; bir dizi talimat; Uzmanlık; kaynak kodu bakımı; makine kodu; işlemek için; tersine mühendislik; atamak; sözdizimi; aletler; sonuç; Yetkin; Uygulama alanı; Merkezi işlem birimi; kodlama temelleri; bir kod parçası; sorumluluk sahibi olmak

Videoyu okumadan önce [Metin](https://www.youtube.com/watch?v=C9FlbhRVYdc&ab_channel=LiveLess) adresinden videoyu izleyin.
konu hakkında bilgi almak için

1. Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Programlama, bir algoritmayı alıp, bir bilgisayar tarafından yürütülebilmesi için bir programlama diline, bir gösterime kodlama işlemidir. Birçok programlama dili ve birçok farklı bilgisayar türü mevcut olmasına rağmen, önemli olan ilk adım, çözüme sahip olma ihtiyacıdır. Algoritma olmadan program olamaz.



Bilgisayar bilimi, programlama çalışması değil ildir. Ancak programlama, bir bilgisayar bilimcisinin yaptığı işin önemli bir parçasıdır.

Programlama, genellikle çözümlerimiz için bir temsil oluşturma şeklimizdir. Bu nedenle, bu dil temsili ve onu oluşturma süreci, disiplinin temel bir parçası haline gelir.

Bilgisayar programlama nedir?

Bilgisayar programlama, belirli bir hesaplama sonucu veya belirli bir görevi gerçekleştirmek için yürütülebilir bir bilgisayar tamamlama ~~programması~~ ve ^{programını} ~~üründürmek~~ ^{üründürmek} ~~ve~~ ^{ve} ~~a~~ ^a

Programlama görevleri ~~incen~~, algoritmalar oluşturmak, algoritmaların doğruluğunu ve kaynak tüketiminin profilini çıkarma ve algoritmaların seçilen bir programlama dilinde uygulanması (genellikle kodlama olarak adlandırılır).

Bir programın kaynak kodu, doğrudan merkezi işlem birimi tarafından yürütülen makine kodu yerine, programcılar anlayabileceğini bir veya daha fazla dilde yazılır. Programlamanın amacı, genellikle belirli bir sorunu çözmek için bir bilgisayarda (bir işletim sistemi kadar karmaşık olabilen) bir görevin performansını otomatikleştirecek bir talimat dizisi bulmaktadır. Bu nedenle, yetkin programlama genellikle uygulama alanı bilgisi, özel algoritmalar ve biçimsel mantık dahil olmak üzere birkaç farklı konuda uzmanlık gerektirir.

Programlamaya eşlik eden ve programlamaya ilişkili eden görevler şunları içерir: test etme, hata ayıklama, kaynak kodu bakımı, yapı sistemlerinin uygulanması ve bilgisayar programlarının makine kodu gibi türetilmiş yapıtların yönetimi. Bunlar, programlama sürecinin bir parçası olarak kabul edilebilir, ancak genellikle yazılım geliştirme terimi, asıl kodun yazılması için ayrılmış programlama, uygulama veya kodlama terimleriyle bu daha büyük süreç için kullanılır. Yazılım mühendisliği, mühendislik tekniklerini yazılım geliştirme uygulamalarıyla birleştirir. Tersine mühendislik, tasarımcılar, analistler ve programcılar tarafından anlamak ve yeniden yaratmak/yeniden uygulamak için kullanılan ilgili bir süreçtir.

kodlama nedir?

Kod, bilgisayarların anlamak ve işlemek için kullandığı dildir.

bizim isteklerimiz.

Kodlama, modern dünyamızın işleyişi iç in hayatı ö neme sahiptir, ancak birçok insan bunun farkında deň ildir. Herhangi bir web sayfasındaysanız, fare altlığı iniza sağ tıklayın ve 'Sayfa Kaynağı'ını Görüntüle'ye tıklayın, ilerleyin ve herhangi bir şey anlayıp anlayamadığ iniza bakın. Orada web sayfası hakkında muhtemelen bilmediğ iniz pek çok bilgi var, ancak bu, kodlama temelleri (HTML, CSS, vb.) ile ilgili ilk deneyiminiz.

Techopedia, kodlamayı "bir şeye bir kod veya sınıflandırma atamak" olarak tanımlar. Bu aslında insanların makinelerle iletişim kurma şeklidir. Ve her şey 1950'lerde, kodlama dillerinin icadı ve gelişimi tüm hızıyla devam ederken başladı. FORTRAN, LISP ve COBOL gibi o zaman oluşturulan kodlama dillerinin çoğu u bugün hala kullanılmaktadır.

Kodlama ve programlama arasındaki temel farklar

Kodlama ve programlama, yazılım geliştirmeye nasıl uyar? endüstriler? İ kisi arasında bilinmesi gereken birkaç önemli fark vardır:

Gerekli beceriler: Kodlayıcıların mutlaka programlama konusunda yetenekli olmaları gerekmeyen, ancak programcılar büyük resim hakkında bilgili olmalı ve kodlamayı anlamalıdır. Programcılar sadece kod yazmakla kalmaz, aynı zamanda yazdıkları kodun mümkün olan en iyi şekilde optimize edilmesini sağ lamanın algoritmaları da anlamak zorundadırlar.

Zorluk: Kodlayıcı olmak istiyorsanız, programlama dili hakkında her şeyi ö ğ renmeniz gereklidir. Bir programcı olarak bunu ve çeşitli algoritmaların nasıl çalıştığı inizi bilmeniz gereklidir.

Çalışma kapsamı: Bir kodlayıcı olarak yazılımı bir araya getirirken, programın bir parçası iç in belirli bir kod parçasını bir araya getirmekten siz sorumlusunuz. Bir programcı olarak, daha büyük resme bakmanız ve yazılımin her yönünden sorunsuz çalışmasını sağ lamanız gerektiğ inden daha fazla sorumluluğ unuz vardır.

Hackr.io'dan alınan bu çizelgedeki bilgiler harika bir kodlama ve programlama arasındaki temel farkların özeti.

Anahtar noktaları	kodlama	Programlama
Yetenekler	Bir dizi talimatı bilgisayarın anlayabileceğ i bir dile dönüştürme işlemidir.	Daha geniş bir kapsamı vardır, bu nedenle kodlamanın yanı sıra gereksinimleri tanımlamayı, sözde kod yazmayı, çalıştırılabilir dosyaları test etmeyi ve oluşturmayı da içerir.

Dürbün	Bir kodlayıcı olarak, programlama dilinin sözdizimini bilmeniz gereklidir.	Bir programcı olarak, kodlama becerileri dışında üst düzey düşünme ve analitik becerilere ihtiyacınız var.
Aletler	Tutulma, Önyükleme, Delphi, ATOM ve daha fazlası	Kodlama araçlarına Git ve Github gibi diğer araçları eklemek için Veritabanı Araçları, Apache Spark gibi Analitik araçlar, Sunum araçları, Bulut araçları da gereklidir.
Sonuç	Çalışan bir kod parçası	Tüm uygulama, bir yazılım ürünü veya bir web sitesi
Destek	Kapsamlı topluluk mevcut geliştirici destektir	Kapsamlı topluluk desteği mevcuttur

[<https://www.lighthouselabs.ca/en/blog/coding-vs-programming>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Algoritmayı bir notasyona kodlamak; bir bilgisayar tarafından gerçekleştirilebilir; dilsel temsil; program kaynak kodu; için uygun anlayış; Tecrübeli; kaynak kodu bakımı; tersi (ters) komutlar; ~~kaynak kodu~~ görüntüleyen, bugün hala kullanılıyor; buluş; sorumlu olmak yürütülebilir kodların oluşturulması; kodlama becerilerine ek olarak; topluluk desteği.

1.2. -al, -able, -ory, ve vb. ekleri kullanarak aşağıdaki fiillerle ilgili sıfatları verin. Bunları kendi cümlelerinizde kullanın

oluşturmak
için programa
değ iştirmek için
düzenlemek

yapıya
değ er vermek
kontrol etmek
için harcamak

1.3. Terimleri tanımlarıyla eşleştirin

1. API	a) dilbilgisi ve imla için bir dizi kural. Başka bir deyişle, bir bilgisayarın yorumlayabileceğ i karakter yapılarını kullanmak anlamına gelir. b)
2. Yazılım	bilgisayarın okuyup yorumladığ i ikili rakamlar veya bitler topluluğ u. c) kullanıcının bir bilgisayarla,
3. Sözdizimi	donanımıyla etkileşime girmesine veya görevleri gerçek ekleştirmesine olanak tanıyan bir talimatlar topluluğ u. d) bir bilgisayar programını analiz
4. Makine kodu	etme ve mantıksal veya sözdizimsel hatalarını ortadan kaldırma süreci.
5. Uygula 6.	e) Başarıyla başarmak veya tamamlamak
Gerçekleştir 7.	f) Yürürlüğ e koymak g) Kuruluşun
Hata Ayıklama	gerektirdiğ i bilgileri içeren bir belge bir pozisyon, mali hibe veya başka bir sınırlı başvuru için bir kişi veya kuruluş kaynak.
8. Uygulama	h) için bir dizi rutin, protokol ve araç yazılım uygulamaları oluşturmak. API'ler, programcılar başka bir şirketin programına veya hizmetine daha kolay girmesini sağ lar.

1.4. Aşağı idaki soruları cevaplayın:

- 1) Kodlama ve programlama arasındaki temel farklar nelerdir?
- 2) Yeterli programlama ne gerektirir?
- 3) Programlamaya eşlik eden ve bununla ilgili görevler nelerdir?
- 4) İnsanlar modern hayatımızda programlamanın önemini fark ediyor mu?
dünya? Neden? Niye?
- 5) Kod programı nasıl yazılır?
- 6) Kapsamlı topluluk desteği i ile ne demek istiyorsunuz?

1.5. Aşağıdaki pozisyonlar için profillerden birini tamamlayın. Bazı ifadeler kullanabilirsiniz, örneğin:

Mezun oldum ... İlk işim izlemeyi iç eriyordu ...

ve ...

Ben ... operasyonlardan sorumluyum. İlk ş, yönetimi de içerir.

Geçmişim... Benden etkilendim... Her zaman büyülendim...

Yurtdışında çalışmak benim için de büyük bir öncelik ve tercih etmemin ana sebeplerinden biri de bu...

İşte bu yıl ... için ... yıllara kaydırıyorum. Bir ofiste olacağımda şimle ilgili bir başka harika şey de şu ki...

Profesyonel olarak, bir kadın olduğunun en az onun kadar iyi olduğunuzu kanıtlamanız gereklidir. bir adam.

... çünkü pek çok yön teknolojiye bağlıdır.

...'daki eğitimin dönenimden sonra, yapmayı planlıyorum ...

Başka bir ... yıl, Paris'e döndüm ve yıllarca tasarım üzerinde çalıştım....

bilgisayar ve IS Yöneticisi	sistem yönetici	Bilgisayar programları	Veritabanı yöneticisi

2. Dilbilgisine Odaklanın

2.1. Cümleleri fiillerin doğru zaman biçiminde tamamlayınız.
parantez

a. Aktif ses 1.

Programı (test etmek) ve hataları (tespit etmek) yarın saat 15.00'e kadar.

2. Bu şirket (oynamak için) kuruluşundan bu yana multimedya geliştirmede önemli bir rol oynamaktadır. 3. Sorunu asla çözmez (yapamaz).

4. Henüz güncellemeleri (yüklememek) için. 5. İnternette hiç televizyon izlediniz mi? 6.

Gelecek yıla kadar bazı üst düzey bilgisayar dillerini (çalışmak için) yapacak. b. Pasif Ses 1.

Programdan sonra (geliştirilecek) (yayılacak) güncel bir versiyon olarak. 2. Programlama dilleri (evrilecek) ile ilgili tüm makaleler

gelecek cuma. 3. Yakın zamanda büyük şirketler için (kurulacak) beş ağ .
4. Dün saat 15.00'e kadar bir akış şeması (tasarlanacak). 5. Program zaten (çevrilecek) makine diline.

2.2. Aşağı idaki cümleleri Aktif Sesteki fiillerle Pasif'e dönüştürün

- 1) Bir program yaptıktan sonra onu kaydedebilir ve arkadaşlarınızla ve ailinizle paylaşabilirsiniz.
- 2) Son olarak, hesaplamayı yaptıktan sonra sonucu ekranda görüntülemek istiyorsunuz. ekran.
- 3) Bilgisayar programcılar, bilgisayarların ve bilgisayar ağlarının bir dizi belirli görevi yerine getirmesini sağ lamak için bilgisayarlar, uygulamalar ve diğer sistemlerle iletişim kurmak için özel diller kullanır.
- 4) George, öğrencilerin çevrimiçi laboratuvarlarda yazılım geliştirdiği ini kaydetti.
- 5) Programcılar, Assembly dilini makine diline kolayca çevrilebilecek şekilde tasarlar.

- 6) Hollandalı programcı Guido van Rossum açık kaynağı geliştirdi
1991 yılında Python dili.
- 7) Başka bir yaklaşım, Web komut dosyaları için tasarlanmış bir dil kullanmaktadır.
tarayıcı yürütür.

2.3. mastar haline getirin. (Bazıları sürekli, mükemmel veya Pasif olabilir). Mastarın formu hakkında yorum yapın. Şu fiilleri kullanın: al, konuş, müdahale et, tasarla, yap, tart, konuş.

- 1) İ şlevler hakkında _____ çok kolaydır, çünkü bunlar somuttur.

- 2) Bize olanak _____ raporlar bize ait.
sağ layacak 3) Aklında bir şey gibi görünyordu.
- 4) Bu yeni programda _____ hiç bir şey yok.
- 5) Başkalarının içinde _____ ne kadar nefret ettiğini biliyorsun.
- 6) Çok çekingendi 7) _____.
Sessizce yenilgisini _____ gibi görünyordu.

3. Tartışma

3.1. Aşağıdaki soruları tartışınız

- 1) "İyi bir programcı" ifadesi sizin için ne ifade ediyor?
- 2) İyi bir programcının sahip olması gereken yetenekler/beceriler nelerdir?
- 3) API veya kütüphane? Hangisi daha önemlidir?

https://www.youtube.com/watch?v=OVvTv9Hy91Q&ab_channel=SimplyExplained 4)

Ana programlama alanlarına bazı örnekler verin.

- 5) Hangi programlama dilleri şimdi önemlidir?

3.2. Partnerinizle yazılım geliştirme hakkında bazı harika Programlama Sözleri tartışın (Jeremy Morgan #programming tarafından)

"Programlama ne bildiğinizle ilgili değil, ne yapabildiğinizle ilgiliidir.
çözmek." - Chris Pine

Özellikle yeni başlayanlar için önemlidir. İlk başta, her şeyi, özellikle de dil sözdizimini bilmek konusunda çok endişeliyiz. Problem çözme, en çok kullandığımız beceridir.

"Yeni bir programlama dili öğrenmenin tek yolu, içinde program yazmaktır." - Dennis Ritchie

Programcılar çoğu unlu "yaparak öğren" türleridir. Hiçbir akademik çalışma veya başka insanların kodlarını izlemek, bir editörü açıp hata yapmaya başlamakla karşılaşamaz.

"Bazen bir şeyi olduğum gibi bırakmak, duraklatmak daha iyidir ve bu programlama için çok doğrudur." - Joyce Wheeler

Geliştiricileri yönetirken, bir sorununuz olduğunda her zaman kalkıp bilgisayardan uzaklaşmanızı tavsiye ederim. En iyi çözümlerinizden bazıları, makinenin başında değil ilken size gelecektir.

"Bazı açılardan programlama resim yapmak gibidir. Boş bir tuval ve bazı temel hammaddeyle başlarsınız. Onlarla ne yapacağınızı belirlemek için bilim, sanat ve zanaatın bir kombinasyonunu kullanırsınız." - Andrew Hunt

Yabancılar programmanın sanat olup olmadığıını sorguluyor. Programcılar yapmaz.

"Test başarısızlığı, başarısızlık ise anlamaya yol açar." - Burt Hapishane

Bazı geliştiriciler testten nefret eder. Ancak tutumunuza değil istirmek ve benimsemek sizin daha iyi bir geliştirici yapar.

"En iyi hata mesajı hiç görünmeyendir." - Thomas Fuchs

Bir dahaki sefere harika bir hata raporlamaya odaklanmaya karar verdiği inizde bunu hatırlayın. "Dildeki en zarar verici söz.. hep yapılmıştır.

bu yoldan" - Grace Hopper

Uzun vadeli bir programcı olmak için değil işimi sevmelisiniz. Sadece tahammül edemezsin, sevmek zorundasın.

4. Ek okuma

4.1. Metni kendinize okuyun ve konuyu ne kadar iyi anladığınızı görmek için aşağıdaki soruları yanıtlayın.

Kuantum fiziğinin ilkelerine göre çalışan bu bilgisayarlar muazzam bir potansiyele sahip olduğunu undan, kuantum bilişim son on yılda artan bir ilgi görüyor. Günümüzde çoğu araştırmaçı, bu bilgisayarların bir gün belirli problemleri klasik bilgisayarlardan daha hızlı çözebileceklerine inanıyorlar, çünkü hesaplamalarını yapmak için belirli bir zamanda çeşitli bilgi bitlerinin üst üste geldiği dolanık kuantum durumları kullanıyorlar. Bu, gelecekte kuantum bilgisayarların klasik bilgisayarların çözemediği sorunları makul bir zaman diliminde verimli bir şekilde çözebilecekleri anlamına geliyor.

Bu kuantum üstünlüğüünün hala kesin olarak kanıtlanması gerekiyor. Bununla birlikte, son zamanlarda bazı önemli teknik ilerlemeler elde edilmiştir. 2019 yazının sonlarında, bir kuantum bilgisayar, en hızlı klasik bilgisayardan daha hızlı bir şekilde - çok spesifik olsa da - bir sorunu çözmeye başardı.

Bazı "kuantum algoritmaları", yani hesaplama stratejileri için, kuantum bilgisayarların potansiyelinden yararlanmayan klasik algoritmaların daha hızlı oldukları da bilinmektedir. Ancak bugüne kadar, bu algoritmalar hala mevcut kuantum donanımında hesaplanamıyor çünkü kuantum bilgisayarlar şu anda hala çok fazla hataya açık.

Kuantum hesaplamanın potansiyelini kullanmak, yalnızca en son teknolojiyi değil, aynı zamanda kuantum algoritmalarını tanımlamak için bir kuantum programlama dilini de gerektirir. Prensipte bir algoritma, bir problemi çözmek için bir "reçete"dir; bir programlama dili, bir bilgisayarın gerekli hesaplamaları yapabilmesi için algoritmayı tanımlar.

Bugün, kuantum programlama dilleri belirli donanımlarla yakından bağ lantılıdır; başka bir deyişle, altta yatan devrelerin davranışını tam olarak tanımlarlar. Programcılar iç in bu "donanım tanımlama dilleri" hantaldır ve hataya açıktır, çünkü bireysel programlama talimatları son derece ayrıntılı olmalıdır ve bu nedenle kuantum algoritmalarını uygulamak iç in gereken ayrıntıları açıkça tanımlamalıdır.

Bilgisayar bilimcileri, belirli bir bilgisayar türünün teknik ayrıntılarından soyutlanan bilgisayar dillerini üst düzey programlama dilleri olarak adlandırır. Silq, kuantum bilgisayarlar iç in ilk üst düzey programlama dilidir. Üst düzey programlama dilleri daha anlamlıdır, yani karmaşık görevleri ve algoritmaları bile daha az kodla tanımlayabilirler. Bu, onları programcılar iç in daha anlaşılır ve kullanımı daha kolay hale getirir. Farklı bilgisayar mimarileri ile de kullanılabilirler.

[Kısaltılmış

<https://www.sciencedaily.com/releases/2020/06/200615115820.htm>]

- 1) Kuantum hesaplama potansiyelini kullanmak iç in a) en son _____. teknoloji b) bir kuantum programlama dili ve en son teknoloji c) bir kuantum programlama dili gereklidir.

- 2) Üst düzey programlama dilleri, a) daha az kodlu _____. karmaşık görevleri ve algoritmaları tanımlayabilir. b) çok kodlu karmaşık görevler ve algoritmalar c) karmaşık algoritmalar

- 3) Bugün, _____ belirli bir donanıma yakından bağ lıdır. a) yazılım programları b) kuantum programlama dilleri c) yüksek seviyeli diller

- 4) Programcılar iç in bu "donanım tanımlama dilleri" _____.
a) kolay yö netilebilir ve hataya açık
b) kolay yö netilemez ve hata c) hantal
ve hataya açık

- 5) Bilgisayar bilimcileri, belirli bir bilgisayar türünün teknik detaylarından soyutlanan bilgisayar dillerini a) yüksek seviyeli programlama dilleri b) düşük seviyeli programlama dilleri c) yüksek seviyeli ve düşük seviyeli programlama dilleri olarak ifade eder.

4.2. Aşağı idaki metni okuyun ve çevirin

Beyin bilgisayar programcılığı için nasıl programlanır?

Dünyanın dört bir yanındaki ülkeler, bilgisayar bilimi öğrencilerinin sayısında bir artış görüyor. ABD ve Kanada'daki ilgili üniversite programlarına kayıtlar 2006-2016 yılları arasında üç katına çıktı ve Avrupa'da da artan sayılar görüldü. Aynı zamanda, birçok farklı ülkedeki hükümetler K-12 bilgisayar bilimi eğitiminin zorladığını içine kodlamaya başlama yaşı giderek genleşiyor. Bilgisayar programlamanın artan popüleritesine rağmen, beynimizin bu nispeten yeni aktiviteye nasıl uyum sağladığını hakkında çok az şey biliniyor. Japonya'daki araştırmacılar tarafından yapılan yeni bir araştırma, farklı uzmanlık seviyelerine sahip otuz programcinin beyin aktivitesini inceleyerek, uzman programcinin beynindeki ön, parietal ve temporal kortekslerin yedi bölgesinin programlama için ince ayarlı olduğunu buldu. Bulgu, daha yüksek programlama becerilerinin, çoklu dağıtılmış beyin bölgelerinden oluşan bir ağ üzerinde ince ayarlanmış beyin aktiviteleri üzerine inşa edildiğini göstermektedir.

Nara Bilim Enstitüsü'nde doçent olan Takatomi Kubo, "Birçok araştırma, uzman ve acemi programcılar arasında davranışsal performans, bilgi yapısı ve seçici dikkat konularında farklılıklar olduğunu bildirdi. Bilmediğimiz şey, bu farklılıkların beyinde nerede ortaya çıktığıdır" diyor. Teknoloji, Japonya ve çalışmanın baş yazarlarından biri.

Bu soruyu yanıtlamak için araştırmacılar acemi, deneyimli ve uzman programcı gruplarını gözlemlediler. Programcılara, işlevsel MRI (fMRI) gözlemi altındayken 72 farklı kod parçasının gösterildi ve her parçacığın dört işlevsel kategoriden birine yerleştirilmeleri istendi. Beklendiği gibi, daha yüksek becerilere sahip programcılar, snippet'leri doğrudan bir şekilde kategorize etmede daha iyidi. Sonraki bir ışıldak analizi, yedi beyin bölgesindeki bilgi miktarının programcinin beceri düzeyi ile güçlendiğini ortaya çıkardı: iki taraflı alt ön gyrus pars triangularis (IFG Tri), sol alt parietal lobül (IPL), sol supramarginal gyrus (SMG), sol orta ve alt temporal gyrus (MTG/IT) ve sağ orta frontal gyrus (MFG).

Kubo, "Uzman programcıların beyinlerinde bu özellikleri belirlemek, programlama uzmanlığıının arkasındaki bilişsel mekanizmaları anlamak için iyi bir başlangıç noktası sunuyor. Bulgularımız, programlama uzmanlığını oluştururan potansiyel bilişsel işlevler kümесini aydınlatıyor" diyor.

Daha spesifik olarak, sol IFG Tri ve MTG'nin doğa dil işleme ve özellikle hedef odaklı bir şekilde anlamsal bilgi alımı ile ilişkili olduğu bilinmektedir. Sol IPL ve SMG, epizodik bellek alımı ile ilişkilidir. Doğa ru MFG ve IFG Tri, uyaran odaklı dikkat kontrolü ile işlevsel olarak ilişkilidir.

"Programlama, insanlık tarihinde nispeten yeni bir faaliyettir ve mekanizma büyük ölçüde bilinmemektedir. Aktiviteyi diğer iyi bilinen insan bilişsel işlevlerine bağlamak, programlama uzmanlığı ve anlayışımızı geliştirecektir. Programlama uzmanlığı hakkında daha kapsamlı bir teori elde edersek, daha iyi sonuçlara yol açacaktır. bilgisayar programlamayı öğrenmek ve öğrenecek için yöntemler" diyor Kubo.

4.3. Metnin kısa bir özetini (Görev 4.2) İngilizce olarak yazın

4.4. Aşağıdaki cümleleri çevirin

1. Teknisyenler-programcılar bilgisayar merkezlerinde, BT şirketlerinde, bankalarda, eğitim kurumlarında çalışır. Bunlar yazılım geliştirme, bilgisayar donanımının sorun giderme, donanım ayarlama, kullanıcı eğitimi ile uğraşmaktadır. 2. Programlama dillerin kullanımına dayanır

kaynak metinlerin yazıldığı programlama programları.

3. Olasılık hakkında karar verildikten sonra görevin yazılım uygulaması, gerekli çözümek için bir algoritma oluşturun. 4. Bir programcının mesleğini seçerken, hazırlıklı olmalısınız. çalışmaların ne üniversiteden sonra ne de üniversiteden sonra bitmeyeceği gerçek eğitimi yüksek bir pozisyon almak. Bu özellik ilk bilgi küresinin olması nedeniyle kuyruk görünür teknoloji oldukça genç ve sürekli gelişiyor. 5. Programlama, makineye neyin, hangi biçimde olduğunu bir açıklamadır. ve kullanıcıya nasıl ulaşılacağı. Yani bir tür insan arzularını makine diline çevirme sanatı.

ÜNİ TE 5. Program grammy ve ölçüleri

Öğrenim hedefleri,

icin MMing I diprogrammühendisligi ariac alt yapisinda kisted hizle fakultet daliyenda olarak es
icinde temel bilgileri edinir I

dilleri
profesyonel pratik anlamak

eşek grubu

c bilgisayarda dil e

r

sözcüklerin anlayışını ve onları anlamanı bekleyenlerin anıtlar
Rusça eşdeğ eakensler ve unutmayın
sözcük öbeklerini verin, anlam

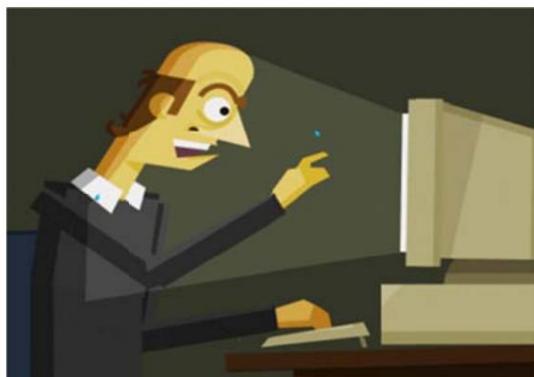
ve

Okumadan önce <https://www.youtube.com/watch?v=www.sen>

metin w izle t ?v=EGQh h5SZctaE E&ab_cm
hannel=C Cok eski ve klasik bir bilgi.

100-10

Aşağıdakileri okuyup **ve** verilen metin egezersizlerini yaptıktan sonra



ö negríam mähengd is rhekkla lairale roit.

- Bu apf yazarlarla ilgili sayarla ve donanıtlı kılınabileceklerini söylenen tarzı atılırlar.

r

ve

Programlama dilleri, düşük seviyeli diller veya yüksek seviyeli diller olarak sınıflandırılabilir. Düşük seviyeli programlama dilleri veya makine dilleri, programlama dillerinin en temel türüdür ve doğrudan bir bilgisayar tarafından anlaşılabilir. Komutlar bit adı verilen 1'ler ve 0'lar dizisi olarak yazıldığında, doğrudan makine dilinde programlamak son derece sıkıcıdır. Assembly dilleri, makine dili programlarının yazılmasını kolaylaştırmak için kullanılır. Örneğin, derleme dilleri, talimatları temsil etmek için ADD, SUB, MPY gibi kısaltmalar kullanır. Program daha sonra bir assembler adı verilen yazılım tarafından makine diline çevrilir.

Assembly dili, makine diline kolayca çevrilebilecek şekilde tasarlanmıştır. Veri bloklarına makine adresleri yerine adlarıyla başvurulmasına rağmen, montaj dili karmaşık bilgileri organize etmek için daha karmaşık araçlar sağlar. Makine dili gibi, montaj dili de dahili bilgisayar mimarisi hakkında ayrıntılı bilgi gerektirir. Bir bilgisayarı çevresel aygıtlarla (yazıcılar, tarayıcılar, depolama aygıtları vb.) etkileşime girecek şekilde programlamak gibi, bu tür ayrıntılar önemlidir.

Üst düzey diller, insan dilinden sözcükleri ve söz dizimini kullanan nispeten karmaşık ifadeler kümesidir ve bu nedenle okunması, yazılması ve bakımı daha kolaydır. Üst düzey diller örnek olarak Pascal (başlangıç veya öğretime dili olarak yaygın olarak kullanılır), C (sistem yazılımı, grafikler ve ticari programlar yazmak için kullanılır), C++ (öncelikle sistem/uygulama yazılımı, sürücüler, istemci-sunucu ile kullanılır) verilebilir. uygulamaları), Cobol (iş uygulamaları için popüler), Fortran (bilimsel ve matematiksel uygulamalar için kullanılır), Java (Web üzerinde çalışmak üzere tasarlanmıştır), Visual Basic (Windows uygulamaları oluşturmak için kullanılır) ve aşağıda gibiler gibi kabuk komut dosyası dilleri UNIX, Linux ve Mac OS X ortamı. Web belgeleri oluşturmak için kullanılan dillerde biçimlendirme dilleri denir, metin dosyalarını biçimlendirmek ve bağlamak için yonergeler (işaretler) kullanırlar, örneğin HTML (Köprü Metni İşaretleme Dili).

Hangi dili kullanırsınız kullanın, bir bilgisayarın anlayabilmesi ve işleyebilmesi için onu makine diline çevirmeniz gereklidir. Bunu yapmanın iki yolu vardır: programı derlemek ve programı yorumlamak.

Derlenmiş bir dilde, programcı daha genel talimatlar yazar ve bir derleyici (özel bir yazılım parçası) bu yüksek seviyeli talimatları otomatik olarak makine diline çevirir. Makine dili daha sonra bilgisayar tarafından yürütülür. Günümüzde kullanılan yazılımların büyük bir kısmı,

bu şekilde programlanmıştır. Yorumlanmış bir programlama dilinde, programcının yazdığı ifadeler program çalışırken yorumlanır. Bu, anında makine diline çevrildikleri ve ardından program çalışırken yürütüldükleri anlamına gelir.

İnsanlar komutları bilgisayara programlama dillerinde iletilerler ve dil seçimi bilgisayarın türüne, programın türüne, programcının uzmanlığıına vb. bağlıdır [10].

Binlerce farklı programlama dili oluşturuldu ve her yıl daha fazlası oluşturuluyor. Birçok programlama dili zorunlu bir biçimde (yani, gerçekleştirecek bir işlemler dizisi olarak) yazılarken, diğer diller bildirim biçimini kullanır (yani, istenen sonuç belirtilir, nasıl elde edileceği değil).

Bir programlama dilinin tanımı genellikle sözdizimi (form) ve anlambilim (anlam) olmak üzere iki bileşene ayrıılır. Bazı diller bir belirtim belgesiyle tanımlanır (örneğin, C programlama dili bir ISO Standardı tarafından belirtilir), diğer diller (Perl gibi) referans olarak kabul edilen baskın bir uygulamaya sahiptir. Bazı diller, bir standart tarafından tanımlanan temel dil ve baskın uygulamadan alınan uzantılar olmak üzere her ikisine de sahiptir.

yaygın.

Programlama dili teorisi, programlama dillerinin tasarımları, uygulanması, analizi, karakterizasyonu ve sınıflandırılması ile ilgilenen bilgisayar biliminin bir alt alanıdır.

[https://en.wikipedia.org/wiki/Programming_language]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Kullanıcı etkileşimi; direkt olarak; kısaltma; denilen bitler; makine diline çevir; karmaşık

bilgi; bilgisayarın iç mimarisi; ifade; içinde

ilk dönüş; hiper metin biçimlendirme dili; derlenmiş dil; yorumlanmış dil; büyük bir yazılım payı; hemen, "anında"; programcı deneyimi; normatif belge; referans açıklaması; dizin.

1.2. Aşağı İdaki kelimeleri tanımlarıyla eşleştiriniz

1. Sözdizimi	a) yürütülebilir bir programın oluşturulması derlenmiş bir programlama dilinde yazılmış kod.
2. Yüksek Düzeyli Dil b)	dönüştürmek veya tercüme etmek için kullanılan bir program Montaj koduyla insanlar tarafından yazılan programların bilgisayar tarafından anlaşılabilen makine koduna (binary) dönüştürülmüşidir. c) dilbilgisi
3. Derleme	ve yazım kuralları kümesi. İçinde başka bir deyişle, bir bilgisayarın yorumlayabileceğin karakter yapılarını kullanmak anlamına gelir. d) Kolay
4. Bir komut dosyası veya komut dosyası dil	anlaşılır anahtar kelimelerden oluşan, bir sayfanın genel görünümünü ve içeriği verileri biçimlendirmeye yardımcı olan adlar veya etiketler. Örneğin BBC, HTML, SGML ve XML. e) bir
5. Montajçı	bilgisayar programlama dilidir. bilgisayarla sınırlı değildir, belirli bir iş için tasarlanmıştır ve anlaşılması daha kolaydır. Daha çok insan diline ve daha az makine diline benzeyen. f) dizisi olan bir bilgisayar dilidir
6. İç şartleme dili	derlenmeden yürütülebilen bir dosya içindeki komutların sayısı. Örneğin Perl, PHP ve Python, JavaScript.

1.3. İngilizce-Rusça anlamca eşdeğer kelimelerin çiftlerini oluşturun:

1. Zaman alıcı, 2. Sofistike, 3. Çalıştırmak, 4. İç stemci-sunucu uygulaması,
5. çevre, 6. sıkıcı, 7. tedavi etmek.

- 1) yüksek karmaşıklık, 2) yorumlama, 3) zaman alıcı,
4) sıkıcı, 5) yürütme için çalışma, 6) harici cihazlar
sistemleri, 7) istemci-sunucu uygulaması.

1.4. Metni tekrar okuyun ve aşağıda ki ifadelerin doğrusu mu yoksa doğrusu mu olduğunu karar verin.

yanlış

- 1) Bilgisayar programları, bir kullanıcıya bir bilgisayarla nasıl etkileşime gireceğini söyleyen talimatlar topluluğuudur.
- 2) Çalışmamız programlama olmadan çok zahmetli ve kolay olurdu.
- 3) Üst düzey programlama dilleri veya makine dilleri, programlama dillerinin en temel türüdür ve doğrudan bir bilgisayar tarafından anlaşılabilir.
- 4) Assembly dili, makineye kolayca çevrilebilecek şekilde tasarlanmıştır. dil.
- 5) Yüksek seviyeli diller, insan dilinden kelimeler ve sözdizimi kullanan ve bu nedenle okunması, yazılması ve bakımı daha kolay olan nispeten karmaşık ifadeler kümesidir.
- 6) Visual Basic, sistem yazılımı, grafik ve ticari programlar yazmak için kullanılır.
- 7) Derlenmiş bir dilde, programcı daha genel talimatlar yazar ve bir derleyici (özel bir yazılım parçası) bu yüksek seviyeli talimatları otomatik olarak makine diline çevirir.
- 8) Bir programlama dilinin tanımı genellikle sözdizimi (form) ve semantik (anlam) ve bunlar arasındaki bağlarılardan oluşan üç bileşene ayrıılır.

1.5. Aşağıda ki soruları metne göre cevaplayınız

- 1) Hangi programlama dillerini kullanıyorsunuz? Neden onları kullanıyorsun?
- 2) Üst düzey dillerde örnekler nelerdir?
- 3) Assembly dili, dahili bilgisayar mimarisi hakkında ayrıntılı bilgi gerektirmez, değil mi?
- 4) "Programı derlemek" ne demektir?
- 5) Programlama dili teorisi ne ile ilgilenir?
- 6) Bir ifadeye doğrusu noktalama işaretlerini eklemeyi unutursanız ne olur?

2. Dilbilgisine Odaklanın

2.1. Doğru varyantı seçin

algoritmik diller

- 1) Algoritmik diller _____ matematiksel veya sembolik ifade etmek için hesaplamalar.
A) tasarlanmış B) tasarlanmış C) tasarlanmış
- 2) 1972'de Dennis tarafından C programlama dili _____
AT&T Corporation'da Ritchie ve Brian Kernighan
programlama bilgisayar işletim sistemleri.
A) geliştirildi B) geliştirildi C) geliştirildi
- 3) C _____ montaj dili ile tüm
bir bilgisayarın iç mimarisinin özelliklerini.
A) paylaşıılır B) paylaşıılır C) paylaşıılır
- 4) Verileri ve programları yapılandırma kapasitesi
daha küçük birimlerin bileşimi, ALGOL'un kine _____ eşittir.
A) karşılaştırılabilir B) karşılaştırılabilir C) karşılaştırılabilir
- 5) ALGOL'un özyinelemeli alt programları vardı;
_____ bir sorunu bir soruna indirgeyerek çözme için kendilerini çağırırlar.
aynı türden daha küçük bir problem.
A) olabilir B) olabilir C) olabilir
- 6) ALGOL, bir
programlama dili, Backus-Naur Formu, bazı varyasyonlarda
sözdizimini (dilbilgisini) belirtmek için standart bir araç haline geldi.
Programlama dilleri.
A) tarif etmek B) tarif etmek C) tarif etmek
- 7) ALGOL, _____ programının içinde bulunduğu u blok yapısını tanıttı.
hem veri hem de talimat içerebilen bloklardan oluşur ve
bütün bir programla aynı yapıya sahiptir.
A) B) ---- C)
- 8) C, kompakt bir notasyon kullanır ve programcıya _____
veri adreslerinin yanı sıra bunların adresleri ile çalışabilme
değerler.
A) ile B) --- C) için

9) Önemli algoritmik dil, 1957'de bir IBM ekibi tarafından tasarlanan FORTRAN (formül çevirisi) idi.

John Backus.

- A) lider B) C) tarafından yönetilir) 10 tarafından yönetilir) FORTRAN _____ gerçek sayılar ve bunların bir veya çok boyutlu diziler olarak düzenlenmiş koleksiyonları ile bilimsel hesaplamalar için.

- A) alındı B) amaçlandı C) bestelendi

3. Tartışma

3.1. Aşağıdaki soruları küçük veya büyük gruplar halinde tartışın. Gerekirse İnternet kaynaklarını kullanabilirsiniz.

- 1) Bu dilleri seçerken ne gibi zorluklarla karşılaşınız?
 - 2) Arkadaşınız şöyle diyor: "Kodlamayı öğrenmeye yeni başladım ve bazen kavramları hemen anlayacak kadar zeki olmadığıımı hissediyorum. Takılıp kalmayı nasıl önleyebilirim ve materyalleri öğrenmeye ve ilerlemeye devam etmenin en iyi yolu nedir?" Ne öneriyorsun?
 - 3) Programlamada bir sorunu nasıl tanımlarsınız?

3.2. 'Programlama dilleri' bilgisayar teriminin kısa bir açıklamasını yapın

3.3. Sunumlar ve Raporlar için önerilen konular

- 1) Programlama dillerinin türleri.
 - 2) İş odaklı diller – SQL (yapılandırılmış sorgu dili) ve COBOL (ortak iş odaklı dil).
 - 3) Programcıların karşılaştığı temel sorunlar.

4. Ek Okuma

4.1. Aşağıdaki 'Programlama Paradigmaları' metnini okuyun ve çevirin

Programlama dilleri, üzerinde çalışıkları bilgisayarın işlemlerini taklit eder. Bu nedenle tasarılandıkları bilgisayar, programlama dilinin nasıl oluşturulduğu ve hangilerinin geliştirildiği üzerinde önemli bir etkiye sahiptir.

özellikler dile atfedilir. Bir programlama dilinin çeşitli nitelikleri, dilin hesaplama paradigmاسını belirleyecektir. Aşağıda kiler farklı paradigmalarıdır.

Zorunlu Paradigma: Talimatlar sırayla yürütülür, değil işkenler bellek konumlarını temsil etmek için kullanılır ve atamalar, değil işkenlerin değil erlerini değil istirmek için kullanılır. Emir dilleri, komutları temsil eden ifadelerin dizilişi nedeniyle prosedürel diller olarak da adlandırılır. Şu anda kullanılan çoğu programlama dili zorunludur.

Fonksiyonel Paradigma: Matematiğe ve lambda hesabındaki bir fonksiyonun soyut kavramına dayanır. Bu paradigma, hesaplamanın tanımını fonksiyonların değil erlendirilmesine veya fonksiyonların bilinen değil erlere uygulanmasına dayandırır. İşlevsel paradigmayi içeren dillere bazen uygulamalı diller denir. İşlevsel paradigma, programın bir işlevi değil erlendirdiği, değil erleri belirli işlevlere parametre olarak aktardığı ve işlevlerden değil erler döndürdüğü bir işlevsel çağrısı kullanır. LISP, işlevsel bir programlama dilinin bir örneğidir.

Mantık paradigmasi: Mantıksal programlama, sembolik mantığa dayanır. Bu diller, belirli bir şekilde yürütülmek üzere sınırlandırılmış bir dizi cümle vermek yerine, bir ifadenin doğruluğunu tanımlayan bir dizi ifadeye dayanır. Bu dillerin döngülere ihtiyacı yoktur ve tek gereklilik, hesaplamanın özelliklerinin ifadesidir. Tüm özellikler bildirildiğiinden ve yürütme sırası olmadığından, mantıksal programlamaya bildirimsel programlama denir.

Tek yaygın olarak kullanılan mantık tabanlı dil Prolog'dur.

Nesneye Yönelik Paradigma: Bu paradigma, bir nesne fikrine dayanır. Nesneler, bu bellek konumlarının değil erlerini değil istirebilen tüm işlemlerle birlikte bir konumlar topluluğu olarak tanımlanabilir. Bir nesnenin bir örneği bir değil işkendir. Birçok nesne yönelimli dilde nesneler, aynı özelliklere sahip tüm nesneleri temsil eden sınıflara konur. Bu sınıflar dört şeyi tanımlar. İlk olarak, bir yapıçı bellek ayırır ve bir nesnenin verileri için bir başlangıç değil eri sağlar. İkinci olarak, sınıfın ilk kısmından değil ere erişmenin bir yolu belirlenir. Daha sonra prosedürler yürütülür ve bir değil er tanımlanır.

Nesne yönelimli programlama çok sayıda yeni dilde bulunur ve programlamanın geleceği için temel bir unsur gibi görülmektedir.

```
// Purpose: This program find the absolute value of an
integer
//           without using a function
//
#include <iostream>
using namespace std;
int main()
{
    int number;
    int abs_number;
    // Ask for input
    cout << "This program finds the absolute value of an
integer." << endl;
    cout << "Enter an integer (positive or negative): ";
    cin >> number;
    // Find the absolute value
    if(number >= 0)
    {
        abs_number = number;
    }
    else
        abs_number = -number;
    // Print out output
    cout << "The absolute value of " << number << " is "
<< abs_number;
    cout << endl;
    return 0;
}
```

[<http://www.csun.edu/~vgc30838/Projectch.html>]

4.2. Yukarıda verilen metnin kısa bir özetini yazın (Görev 4.1.)

4.3. Aşağıdaki soruları cevaplayın

Kendini test et

Programlama test sorularına giriş

1) Programlama dili nedir?

A) Yazılı İngilizce B)

Bilgisayarı programlamak için kullanılan yapay bir dil C)

Sözde kodda kullanılan bir dil 2) Makine kodu nedir?

A) Bilgisayarın seri numarası B)

Bilgisayarın anladığını bir programlama dili C) Bilgisayarın markası ve modeli 3) Program nedir?

A) Bir bilgisayara nasıl yapılacağıını anlatan bir dizi adım adım talimat
bir görevi çözmek

B) Bilgisayarda izlenen video

C) Bilgisayarda yazılmış bir akış şeması 4)

İfade nedir?

- A) Akış şemasında bir kutu
- B) Bir programlama dilinde bir anahtar kelime
- C) Bir programlama dilinde yapılan hesaplama

5) Talimat nedir?

- A) Akış şemasındaki bir kutu
- B) Bir programlama dilinde yapılan bir hesaplama
- C) Bilgisayara talimat vermek için birlikte grüplendirilmiş bir veya daha fazla ifade bir görevi

gerç ekleştirmek için 6) 'print' ifadesi ne işe yarar?

- A) Yazıcıya bir programın basılı kopyasını yazdırın
- B) Ekrana bir mesaj yazın
- C) Akış şemasının basılı bir kopyasını yazıcıya yazdırın
- D) 7) 'while' ifadesi ne işe yarar?
- E) Programa devam etmeden önce bilgisayara bir süre beklemesini söyleyin
- F) Seçimi uygulayın
- G) Bir döngü uygulayın
- H) 8) 'def' ifadesi ne yapar?

- A) Bir işlev veya prosedür oluşturur
- B) Bir döngü uygular
- C) Seçimi uygular

9) 'if' ve 'else' ifadeleri ne yapar?

- A) Seçimi uygula
- B) Bir döngü uygula
- C) Bilgisayara devam etmeden önce bir süre beklemesini söyle program

10) Bu kod satırında kaç ifade vardır: print ("Eğ er 17, araba kullanabilirim")?

- A) İki ifade vardır - 'print' ve 'if'
- B) Hiçbir ifade yok
- C) Tek bir ifade var - 'yazdır' [https://

[www.bbc.co.uk/bitesize/guides/zts8d2p/test\]](https://www.bbc.co.uk/bitesize/guides/zts8d2p/test)

4.4. Aşağı İdaki cümleleri çevirin

1) Bir programlama dili, tarafından icat edilen bir dizi sözcüksel, sözdizimsel ve anlamsal kurallardır.

insanlar programlar oluşturmak için.

2) 6-10 ayda başlangıç seviyesinde bir dil öğrenebilirsiniz, ancak yanlış seçim yaparsanız, dil günceliğini yitirebilir ve kaybedersiniz. zaman ve para. 3)

Programlama talebi^{İnteract}lamak için şirketler özel

Diller

derecelendirme.

4) C, en eski ve en popüler dillerden biridir.

programlama. "Hafif" ve hızlıdır, bu nedenle yüksek performansın gerekliliği u yerlerde kullanılır. Örneğin, sürücüler, işletim sistemleri veya Mikrodenetleyiciler için yazılım.

5) Java, geniş kapsamlı bir çapraz platform dilidir.

kütüphanelerin sayısı ve geniş bir geliştirici topluluğu u.

6) Çapraz platform, bir program yazabilme yeteneğidir.

bir kez ve hemen birkaç ameliyathanede kullanın sistemler: Windows, Linux ve MacOS.

Java kütüphaneleri sayesinde hemen hemen her şey için uygundur: iş grafik, ses, küçük oyunlar oluşturma.

Görevin anahtar cevapları 4.3.

1. B); 2. B); 3 A); 4. B); 5. C); 6. B); 7.C); 8. A); 9. Bir); 10. C).

ÜNİ TE 6. Nesneye Yönetik Programlama (OOP)

Öğrenme hedefleri

Nesne yönelimli programlama ve içindeki dil türleri hakkında temel bilgi edinmek Nesne yönelimli programmanın temel özelliklerini anlamak OOP ile geleneksel programlama arasındaki farkı düşünmek

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Kendi kendine yeten varlıklar; resmi kurallar dizisi; ilişkisel veritabanı; nesne veritabanı; kapsülleme; miras; kod modülerliği; rutinler; hiyerarşinin en altında; miras kalmak; polimorfizm; entegre olmak; bellek tahsisi; dağ itmek; "olay güdümlü" programlama; başarılı olmak; taşınabilir; veri adresleri; özellik; uzatmak; birbiriyle ilişkili uygulamalar; benzer olmak

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Verilerin ve bunlarla ilişkili işlemlerin ("yöntemler") "nesneler" olarak adlandırılan kendi kendine yeten varlıklar olarak tanımlanmış bir programlama dili yapısı. C++ ve Java gibi bugünün normu olan nesne yönelimli programlama (OOP) dilleri, nesneleri oluşturmak ve yönetmek için resmi bir dizi kural sağlar. Veriler, geleneksel bir ilişkisel veritabanında veya verilerin karmaşık bir yapıya sahip olması durumunda bir nesne veritabanında depolanır. yapı.



Nesne yönelimli programlamada onları OOP olmayan dillerden farklı kılan üç ana özellik vardır: kapsülleme, kalıtım ve polimorfizm.

Kapsülleme Modülerlik Zorlar Kapsülleme, işleme işlevlerini verilere bağlayan bağimsız modüllerin oluşturulmasını ifade eder. Bu kullanıcı tanımlı veri türleri

"sınıflar" olarak adlandırılır ve bir sınıfın bir örneği bir "nesne"dir. Örneğin, bir bordro sisteminde, bir sınıf Yönetici olabilir ve Pat ve Jan, Yönetici sınıfının iki örneği (iki nesne) olabilir. Kapsülleme, rutinleri ayrı tutan ve birbirleriyle çatışmaya daha az eğilimli olan iyi kod modüllerliği sağlar.

Kalıtım Geçişleri "Bilgi" Aşağı Sınıflar,

hiyerarşiler halinde oluşturulur ve kalıtım, bir sınıftaki yapı ve yöntemlerin hiyerarşiden geçirilmesine izin verir. Bu, karmaşık sistemlere işlev eklerken daha az programlama gerektiği anlamına gelir. Bir hiyerarşinin en altına bir adım eklenirse, yalnızca bu benzersiz adımla ilişkili işleme ve verilerin eklenmesi gereklidir. Diğer her şey miras alınır. Mevcut nesneleri yeniden kullanma yeteneği, nesne teknolojisinin önemli bir avantajı olarak kabul edilir.

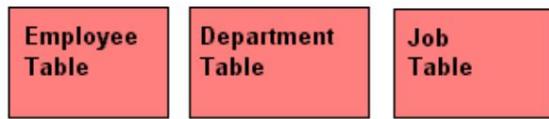
Polimorfizm Herhangi Bir Şekil Alır

Nesne yönelimli programlama, tam türü çalışma zamanına kadar bilinmeyen nesnelerle ilgili prosedürlerin oluşturulmasına izin verir. Örneğin, bir ekran imleci, program moduna bağlı olarak şeklini bir oktan bir çizgiye değistirebilir. Fare hareketine yanıt olarak imleci ekranda hareket ettirme rutini "imleç" için yazılır ve polimorfizm, imlecin çalışma zamanında gerekli olan şekli almasına izin verir. Ayrıca yeni şekillerin kolayca entegre edilmesini sağlar.

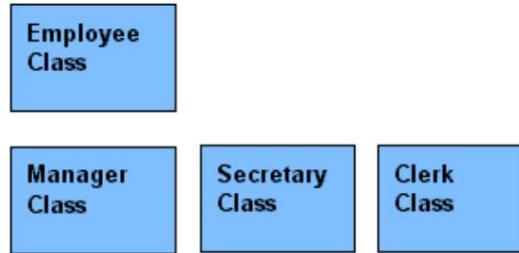
Aşağıda, temel OOP terimlerini geleneksel programlama ile karşılaştırmaktadır.

AÇIK	Geleneksel Programlama
sınıf	veri + işlemeyi tanımla
nesnesi	veri + veri işleme (bir alan) işlev işlev
öznitelik	alan) işlev işlev
yöntemi	çağrısı bir yapı
mesaj	tahsis eder
somutlaştırma	

Relational modeling



Object modeling



Şekil 9. İlişkisel ve Nesne Modelleme

Aynı çalışan, departman ve iş tabloları yerine, bir çalışan sınıfı tüm çalışanlar için verileri ve işlemleri içerir. Her alt sınıfın (yönetici, sekreter vb.) kendi verisi ve işlemesi vardır, ancak aynı zamanda her şeyi çalışan sınıfından devralır. Çalışan sınıfında yapılan değişiklikler her alt sınıfı etkiler.

OOP Dilleri Bugün,

C++, C#, Java, JavaScript, Visual Basic.NET ve Python popüler nesne yönelimli diller.

1980'lerin ortalarında geliştirilen C++ dili, C programlarının verimliliğiini korurken nesneler ekleyerek C'yi genişletti. Hem eğitimi hem de endüstriyel programlama için en önemli dillerden biri olmuştur. Birçok işletim sisteminin büyük bir kısmı C++ ile yazılmıştır.

Java ile birlikte C++, birbirile ilişkili birden çok uygulamayı içeren ticari yazılım paketleri geliştirmek için popüler hale geldi. C++ en hızlı dillerden biri olarak kabul edilir ve düşük seviyeli dillere çok yakındır, bu nedenle bellek tahsisini ve yönetimi üzerinde tam kontrol sağlar. Bu özellik ve diğer birçok yeteneği, aynı zamanda onu öğrenmesi ve büyük ölçüde ele alınması en zor dillerden biri haline getirir.

C# (müzik notası gibi C keskin olarak telaffuz edilir) 2000 yılında geliştirildi. C#, C ve C++'inkine benzer sözdizimine sahiptir ve genellikle Microsoft Windows işletim sistemi için oyunlar ve uygulamalar geliştirmek için kullanılır.

1990'ların başında Java, Sun Microsystems, Inc. tarafından World Wide Web (WWW) için bir programlama dili olarak tasarlandı. Görünüş olarak C++'a benzemesine rağmen, nesne yönelimli idi. Özellikle Java, manipüle etme yeteneği de dahil olmak üzere daha düşük seviyeli özelliklerden vazgeçti.

veri adresleri, dağ itilmiş sistemler iç in programlarda ne arzu edilen ne de yararlı olan bir yetenek. Java programları taşınabilir olması iç in her bilgisayar platformuna özel bir Java Sanal Makinesi tarafından ç evrilir ve ardından Java programını çalıştırır. Java, Web "applet'leri" aracılığ ıyla İ nternet'e etkileşimli yetenekler eklemenin yanı sıra, cep telefonları gibi küçük ve taşınabilir aygıtları programlamak iç in yaygın olarak kullanılmaktadır.

Visual Basic , nesneler ve "olay güdümlü" programlama: düğ meler, menüler ve grafik kullanıcı arabirimlerinin (GUI'ler) diğ er ög eleri ekleyerek BASIC'in yeteneklerini genişletmek iç in Microsoft tarafından geliştirilmiştir. Visual Basic, küçük rutinleri programlamak iç in diğ er Microsoft yazılımlarında da kullanılabilir. Visual Basic, 2002'de, C++ ile benzerlikleri olan bir dil olan C#'a dayalı çok farklı bir dil olan Visual Basic .NET tarafından başarılı oldu.

Açık kaynak dili Python , 1991 yılında Hollandalı programcı Guido van Rossum tarafından geliştirilmiştir. Grup ifadeleri iç in parantez yerine girinti kullanma gibi özellikleri ile kullanımı kolay bir dil olarak tasarılmıştır. Python ayrıca karmaşık işlerin yalnızca birkaç ifadeyle yürütülebilmesi iç in tasarlanmış çok kompakt bir dildir. 2010'larda Python, Java ve JavaScript ile birlikte en popüler programlama dillerinden biri haline geldi.

[[https://www.computerlanguage.com/results.php?definition=nesne
yönelimli+programlama](https://www.computerlanguage.com/results.php?definition=nesne_yönelimli+programlama)]
[<https://www.britannica.com/technology/computer-programming-language/>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İ ngilizce karşılıklarını verin:

Hangi veri ve yöntemlerde; kendi kendine yeterli; veritabanına kaydet veri; özellikler; mülk mirası; örnek; sağ ıamak; kod modüllerliği; bakım; özellik ekleme; yeniden kullanma yeteneği; ana avantaj; fırlatma zamanı; cevap olarak; program modu; etkilemek kaydetmek; kullanımı kolay; girintiler ekleme; onun yerine

parantez; kompakt dil.

1.2. Aşağıdaki ifadeleri doğrudan kelime sırasına koyun. bunları kullanın kendi cümlelerin

- a) bağımsız "nesneler" olarak adlandırılan _____
 varlıklar - b) kullanıcı tanımlı veri türleri - c)
 en alta bir hiyerarşi eklenir - d) bir ok _____
 şeklärinden çizgisine değişir - e) korurken _____
 verimliliği programlar C - f) karmaşık ifadelerle _____
 yürütülen işler yalnızca birkaç olabilir -

1.3. Aşağıdaki kelimeleri tanımlarıyla eşleştiriniz

1. Modülerlik	a) bir dizi ayrı, ilgili dosya (tablo) bulunduran, ancak gerektiğiinde sorgular ve raporlar için dosyalardan veri öğelerini birleştiren bir veritabanı. b) bilgisayarın
2. ilişkisel veritabanı	talimatlarını yerine getirmesini sağlayan bir program çalıştırır.
3. polimorfizm	c) nesne yönelimli bir veritabanı yönetim sistemi (ODBMS) tarafından yönetilen bir veritabanı.
4. nesne veritabanı	d) Nesne teknolojisinde, hem verileri hem de işlemeyi içeren bağımsız modüllerin oluşturulması. 5. kapsülleme e) birbiriyle etkileşen daha küçük alt sistemlere bölünmüş bir sistemin özelliği i.
6. yürütmek	f) birçok şekil anlamına gelir. Nesne teknolojisinde, bir istek (mesaj) gönderildiği nesneye göre farklı sonuçlar ürettiğinde polimorfizm sergilenebilir.

1.4. Aşağıdaki soruları metne göre cevaplayınız

- 1) OOP'de veriler nerede saklanıyor?
- 2) OOP'deki başlıca özellikler nelerdir?
- 3) Kapsülleme neyi ifade eder ve neyi sağlar?
- 4) Sınıflar nasıl oluşturulur?
- 5) C++ ve C# arasındaki fark nedir?

6) Açık kaynak dili Python ne zaman geliştirildi?

7) 'Çok kompakt bir dil' ne anlama geliyor?

1.5. Metni tekrar okuyun ve aşağıdaki ifadelerin doğrusu mu yanlış mı olduğunu karar verin

- 1) C++ ve Java gibi nesne yönelimli programlama dilleri, nesneleri oluşturmak ve yönetmek için resmi bir dizi kural sağlar. Veriler geleneksel bir ilişkisel veritabanında saklanır.
- 2) Nesne yönelimli programlamada onları OOP olmayan dillerden farklı kılan önemli bir özellik vardır. Polimorfizmdir.
- 3) Sınıflar hiyerarşiler halinde oluşturulur ve kalıtım, bir sınıfın yapı ve yöntemlerin hiyerarşiden geçirilmesine izin verir.
- 4) Nesne yönelimli programlama, çalışma zamanına kadar kesin türü bilinmeyen nesnelerle ilgili prosedürlerin oluşturulmasına izin verir.
- 5) 2000'li yıllarda geliştirilen C++ dili, C programlarının verimliliğini korurken nesneler ekleyerek C'yi genişletti.
- 6) 1990'ların başında Java, Sun Microsystems, Inc. tarafından World Wide Web (WWW) için bir programlama dili olarak tasarlandı.
- 7) Visual Basic, diğer Microsoft yazılımlarında da kullanılabilir. küçük rutinler programlayın.
- 8) Java ve JavaScript, cep telefonları gibi küçük ve taşınabilir cihazları programlamak için kullanılmaz.
- 9) 2010'larda Python, Java ve JavaScript ile birlikte en popüler programlama dillerinden biri haline geldi.

2. Dilbilgisine Odaklanın

2.1. Bağlaç zarflarını kullanarak iki bağimsız tümceyi (cumleyi) birleştirebiliriz. Bağlaç zarfları neden ve sonuç, sıra, karşılıklık, karşılaştırma veya diğer ilişkileri gösterir.

Bunlardan en yaygın olanları:

Buna göre	Aslında	Aksi halde
Sonradan	Aynı şekilde	benzer şekilde
Ayrıca	Dahası	Hala
Sonuç olarak	Yine de	Öyleyse
Yine de	Her şeye rağmen	

2.2. Aşağı İdaki kuralları ve örnekleri öğrenin. kendin yap

Yukarıda listelenen bağlaç zarflarından bazılarını içeren cümleler.

Kuralları Öğrenmek: Konjonktif Zarf Örnekleri

Kural 1: Zarfla Bağlantılı Tam Cümleler Noktalı Virgül Gerektirir Cümleyi tamamla +; + zarf bağlantı kelimesi + tam cümle.

Örneğin Jeffrey programlama dillerini öğrenmek istemiyor; yine de annesi onu programlama derslerine yazdırıyor.

Bununla birlikte, yukarıdaki cümlede zarf bağlayıcı kelimedir. Ama, so, ve henüz gibi koordine edici bir bağlantı kelimesiyle aynı şekilde çalışıyor. Bununla birlikte, buradaki fark, eşgüdümlü bir bağlaç, noktalı virgül gerektirmezken, zarf bağlayıcı bir sözcük gerektirir.

Kural 2: Tek Bir Ana Cümle ile Bağlayıcı Zarflar Kullanabilirsiniz Bir ana tümcenin başında, ortasında ve sonunda bağlayıcı zarflar kullanabilirsiniz. İşte bazı örnekler: a) Frank, kablo şirketi tarafından yaklaşık iki saat bekletildi. Sonunda bir müşteri hizmetleri temsilcisiyle temasa geçti. b) Jan, GTA'da hiç yüksek puanı alamadı. Yine de puanını yükseltmeye kararlı. c) Programlama sınavına çalısmakla ilgili onu endişelendiren bir şey vardı. Bununla birlikte, bilişim okumakta zorluk çekmedi.

Kural 3: Cümleye Bağlı Olarak, Bağlayıcı Bir Zarf Kullanırken Virgül Kullanmanız Gerekmezdir Bazen, cümledeki bir boşluk, okuyucuyu virgülle durdurmayı haklı çıkarmak için çok zayıftır. Aslında, virgül bir cümleyi dalgalı yapabilir.

Örneğin Harrison, öğretmeninin sınıfta bir soruyu cevaplaması için onu aramasından kesinlikle hoşlanmadı. Harrison, kesinlikle, öğretmeninin sınıfta bir soruyu cevaplaması için onu aramasından hoşlanmadı.

Bunlar hem akademik hem de profesyonel yazı için önemli becerilerdir.

Aşağıdaki daha birleşik zarf örneklerine bakın. a) Paul sınıf

arkadaşının ödevini kopyaladı. Sonuç olarak, öğretime derecesini tutturdu.

b) Stacy bilgisayar mağazasına gitti; ancak, en sevdiği akıllı telefonun dışındaydılar.

c) Çalışmanız kötü değil; aslında, muhtemelen bir zam hak ediyorsun.

d) Yazar blogunu geliştirdi; dolayısıyla yıllarca başka roman yazmadı. e) Haftalık dersin iptal edildiğiini unutmuşlardır; şüphesiz onlar vardı

boş zamanlarını nasıl geçireceklerini bulmakta zorlanırlar.

3. Konuşma ve Yazma

3.1. İşte Nesne Yönelimli hakkında okumanız gereken bazı harika alıntılar ve sözler.

Bunları partnerinizle tartışın

Nesne yönelimli programlama, spaghetti kodu yazmak için sürdürülebilir bir yol sunar. Programları bir dizi yama olarak toplamanıza izin verir. -

Yazar: Paul Graham

Mac göz kırdı. Bu gülümseme ölümcül bir silah olarak kayıt altına alınmalı. -

Yazar: Kaje Harper

Yardımı hak eden birinden asla uzaklaşma; senin elin o kişi için Tanrı'nın elidir. -

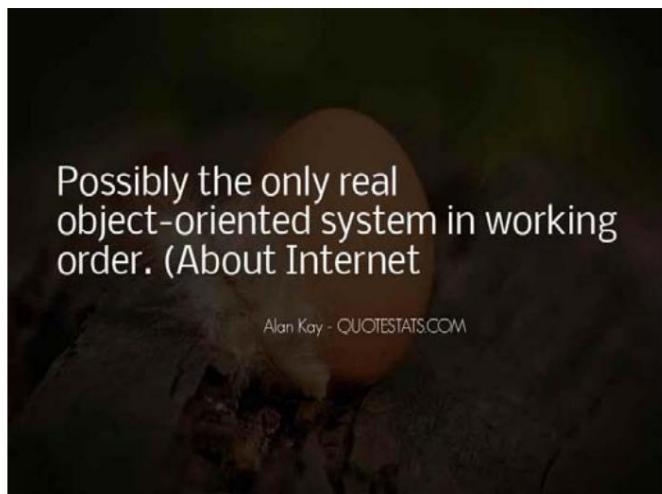
Yazar: Eugene H. Peterson

Simula 67'de ortaya çıktıktan sonra nesne yönelimli programlama, yazılım yapısının gerçek dünya yapılarına dayanmasına izin verir ve programcılara karmaşık programların tasarımını ve inşasını basitleştirmenin güçlü bir yolunu sunar. -

Yazar: David Gelernter

Her bağımızlık, sınıfınızın dokunduğu şeylerle yapışmasına neden olan küçük bir tutkal noktası gibidir. - Yazar: Sandi Metz

'Nesneye Yönelik' terimini ben icat ettim ve size şunu söyleyebilirim ki ben yapmadım. aklında C++ var. - Yazar: Alan Kay



3.2. Nesne yönelimli programlamadaki her özelliği i kendi kelimelerinizle açıklayın

3.3. Sunumlar ve Raporlar için önerilen konular

- 1) OOP'nin faydaları nelerdir?
- 2) OOP'nin Eleştirisi. Nesneye Yönelik Programlamada Yanlış Olan Ne?
- 3) OOP neden bu kadar popüler?
- 4) Nesneye yönelik terminoloji

4. Ek okuma

4.1. Aşağıdaki metni okuyun ve çevirin

Belge biçimlendirme dilleri

Belge biçimlendirme dilleri, basılı metin ve grafiklerin organizasyonunu belirtir. Birkaç sınıfa ayrırlırlar: bir kelime işlemci programı ile aynı işlevleri yerine getirebilen metin biçimlendirme gösterimi, bir yazdırma aygıtı tarafından yorumlanan sayfa açıklama dilleri ve en genel olarak, bir belgenin bölümlerinin amaçlanan işlevini tanımlayan biçimlendirme dilleri.

TeX, 1977-86 yıllarında Stanford Üniversitesi profesörü Donald Knuth tarafından kitaplarındaki matematiksel gösterimin kalitesini iyileştirmek için bir metin biçimlendirme dili olarak geliştirildi. Metin biçimlendirme sistemlerinden farklı olarak

WYSIWYG ("Ne Görüyorsan Onu Alırsın") kelime işlemcileri, düz metin biçimlendirme komutlarını bir belgeye yerleştirir ve bunlar daha sonra dil işlemcisi tarafından görüntülenmek veya yazdırılmak üzere biçimlendirilmiş bir belge üretmek üzere yorumlanır. TeX, italik metni örneğin `\{bu italiktir}` olarak işaretler ve ardından bu italik olarak görüntülenir.

Tex, önceki metin biçimlendirme dillerinin yerini büyük ölçüde aldı. Güçlü ve esnek yetenekleri, bir uzmana yazı tipi seçimi, tablo düzeni, matematiksel gösterim ve bir belgeye grafiklerin dahil edilmesi gibi şeyler üzerinde hassas bir kontrol sağladı. Genellikle yeni bir paragraf başlatmak gibi yaygın işlemler için basit komutları tanımlayan "makro" paketlerin yardımıyla kullanılır; LaTeX yaygın olarak kullanılan bir pakettir. Tex, farklı belge türleri için çok sayıda standart "biçim sayfası" içerir ve bunlar her kullanıcı tarafından daha fazla uyarlanabilir. Bibliyografyaları yöneten ve tüm yaygın bibliyografya stilleri için stil sayfalarına sahip BibTeX ve çeşitli alfabelere sahip diller için TeX sürümleri gibi ilgili programlar da vardır.

PostScript, 1980'lerin başında Adobe Systems Incorporated tarafından Xerox PARC'daki (Palo Alto Araştırma Merkezi) çalışma temelinde geliştirilen bir sayfa tanımlama dilidir. Bu tür diller, belgeleri, bir kişisel bilgisayar tarafından belgeyi kendi ekranında görüntülemek için veya bir yazıcı veya dizgi aygıtlındaki bir mikroişlemci tarafından yorumlanabilecek terimlerle tanımlar.

PostScript komutları, örneğin, metni çeşitli yazı tiplerinde ve boyutlarda tam olarak konumlandırılabilir, matematiksel olarak tanımlanmış resimler çizebilir ve renk veya gölgeleme belirtebilir. PostScript, bir işlem adının argümanlarını takip ettiği, ters Lehçe notasyon olarak da adlandırılan postfix'i kullanır. Bu nedenle, "300 600 20 270 ark stroku" şu anlama gelir: konumda (300, 600) yarıçapı 20 olan 270 derecelik bir yay çizin ("strok"). PostScript bir programcı tarafından okunup yazılabılmasına rağmen, normalde metin biçimlendirme programları, kelime işlemciler veya grafik görüntüleme araçları tarafından üretilir.

PostScript'in başarısı, spesifikasyonunun kamu malı olmasından ve yüksek çözünürlüklü lazer yazıcılar için iyi bir eşleşme olmasından kaynaklanmaktadır. Baskı yazı tiplerinin gelişimini etkilemiştir ve üreticiler çok çeşitli PostScript yazı tipleri üretmektedir.

SGML (standart genelleştirilmiş biçimlendirme dili), biçimlendirme dillerinin tanımı için uluslararası bir standarttır; yani bir üst dildir. İşaretleme, bir metin parçasının işlevini veya nasıl görüntüleneceğini belirten etiket adı verilen notasyonlardan oluşur. SGML vurgular

bir etiketin "<vurgu>" olabileceğ i açıklayıcı işaretleme. Böyle bir işaretleme, belge işlevini belirtir ve bir bilgisayar ekranında ters video olarak, bir dactiloyle altı çizilerek veya dizgi metninde italik olarak yorumlanabilir.

SGML, DTD'leri (belge türü tanımları) belirtmek iç in kullanılır. DTD, belgede hangi öğelerin görünmesi gerektiğ ini (örneğ in, <Title>) belirterek ve bir paragrafin bir tablo girişinde görünebileceği gibi belge öğelerinin kullanımı iç in kurallar vererek, rapor gibi bir belge türünü tanımlar. ancak bir paragraf içinde bir tablo görmeyebilir. İ şaretlenmiş bir metin, bir DTD'ye uygun olup olmadığı ini belirlemek için bir ayrıştırma programı tarafından analiz edilebilir. Başka bir program, bir dizin hazırlamak veya belgeyi yazdırmak üzere PostScript'e çevirmek iç in işaretlemeleri okuyabilir. Yine bir başkası, görme veya işitme engelli okuyucular iç in büyük türde veya ses üretebilir (3400). [<https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>]

4.2. TeX, PostScript, SGML ve özellikleri hakkında konuşun

4.3. Aşağı idaki cümleleri çevirin

1) Nesne yö nelimli programlama sunar

bir programcının mesleğ inde ustalaşmanın yolu. Dan beri bilgisayar programlama metodolojisinin icatları büyümeye uyum sağlayarak önemli ölçüde deň işir.
program karmaşıklığı

2) Programların büyümesiyle birlikte montaj dili icat edildi, böylece

programcı daha büyük ve daha karmaşık iç in sembolik gösterimi kullanan programlar makine talimatları.

3) Sonunda, üst düzey diller tanıtıldı,

programcı sorunu çözmek iç in daha fazla yol program karmaşıklığı İ lk yaygın dil FORTRAN'dı. FORTRAN çok olmasına rağmen etkileyici ilk adım, programların anlaşılırlığ ini ve kolaylığı ini sağlayan bir dil olarak kabul edilemez. 4) Programlamanın geliştirilmesindeki kilometre taşları, artan karmaşıklık sorununu çözmeye hizmet eden yöntemlerdir.

programlar. Bu yolculuğ un her aşamasında, yeni yaklaşım şunları iç erir: önceki yöntemlerin en iyi unsurlarını iç erir.

- 5) Nesne yö nelimli programlama özümsemiştir.
yapilandırılmış programlamanın en iyi fikirleri ve bunları birleştirin
görmek için güçlü yeni konseptlerle
yeni bir ışıkta programlama sorunu. 6)
Nesne yö nelimli programlama kolaylaştırır
Problemi, etkileşen parçaların alt gruplarına ayırin. Daha sonra bu
alt grupları, özellikler adı verilen birimlere dönüştürebilirsiniz.

- 7) Tüm nesne yö nelimli dillerin üç ortak noktası vardır.
kavamlar: kapsülleme, polimorfizm ve kalıtım.

ÜNİ TE 7. Programmanın Unsurları. Kontrol Yapıları

Öğrenme hedefleri

Kontrol yapıları ve veriler arasındaki farkı anlamak

yapılar

Üç temel kontrol yapısı hakkında temel bilgi edinmek Bir alt program
örneğini ve işlevini düşünmek

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Algoritmaları ifade etmek için; prosedürel diller; sekans; yinelemeli; değişkenleri atamak için; kökleri bulmak için; koşullu; ikinci dereceden denklem; döngü; değişken; üstlenmek; ayrımcı; alt program; mutlak değişken fonksiyonu; çarpma işlemi; uygulanacak; tekrarlar; yaklaşım; izlenebilir alt programlar; basit bir yeniden ifade; bir karekök; yürütme yolları; özyinelemeli alt programlar; faktöriyel fonksiyon.

<https://study.com/academy/lesson/5-basic-elements-of-programming.html> adresindeki videoyu okumadan önce.

İşeriğini Rusça olarak işaretleyin

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Gösterim farklılıklarına rağmen, çağdaş bilgisayar dilleri aynı programlama yapılarının çoğuunu sağlar. Bunlar, temel kontrol yapılarını ve veri yapılarını içerir. İki algoritmaları ifade etmek için araçları sağlar ve ikincisi, bilgiyi organize etmenin yollarını sağlar.

Kontrol yapıları En

yaygın tür olan prosedürel dillerde yazılan programlar, içerik listeleri ve bunları kullanmak için adım adım talimatlar içeren tarif gibidir. Hemen hemen her prosedür dilindeki üç temel kontrol yapısı şunlardır: 1. Sıralama—sıvı bileşenleri birleştirin ve ardından kuru malzemeyi ekleyin.

olanlar.

2. Koşullu—domatesler tazeyse kaynatın, ancak konserve ise bu adımı atlayın.

3. Yinelemeli—yumurta aklarını yumuşak tepeler oluşana kadar çırpin.

Sıra, varsayılan kontrol yapısıdır; talimatlar birbiri ardına yürütülür. Örneğin, ikinci dereceden bir denklem ekseninin köklerini bulmak için sonuçları değil işkenlere atayarak bir dizi aritmetik işlem gerçekleştirebilirler.

$x^2 + bx + c = 0$. Koşullu IF-THEN veya IF-THEN-ELSE kontrol yapısı, bir programın alternatif yürütme yollarını izlemesine izin verir. Yineleme veya döngü, bilgisayarlarla güçlerinin çoğuunu verir.

Bir dizi adımı gerektiği kadar tekrarlayabilirler ve oldukça basit adımların uygun tekrarları karmaşık sorunları çözebilir.

Bu kontrol yapıları birleştirilebilir. Bir dizi birkaç döngü içerebilir; bir döngü, içinde yuvalanmış bir döngü içerebilir veya bir koşulun iki dalının her biri, döngüler ve daha fazla koşul içeren diziler içerebilir. Bu makalede kullanılan "sözde kod"da "*" çarpmayı, " " ise değil işkenlere değil er atamak için kullanılır. Aşağıındaki programlama parçası, ikinci dereceden formülü kullanarak ikinci dereceden denklemin bir kökünü bulmak için IF-THEN yapısını kullanır:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

İkinci dereceden formül, a 'nın sıfır olmadığıını ve diskriminantın (kare kök işaretinin içindeki kısım) negatif olmadığıını (gerçek bir sayı kökü elde etmek için) varsayar. Koşullar şu varsayımları kontrol eder:

IF $a = 0$ SONRA

KÖK c/b

BAŞKA

AYIRIMCI $b*b - 4*a*c$

AYIRICI 0 İ SE KÖK ($b +$

KARE_KÖK(AYIRIMCI))/2*a

ENDIF

ENDIF

Yukarıdaki parçada kullanılan SQUARE_ROOT işlevi, bir alt program örneği idir (ayrıca prosedür, alt program veya işlev olarak da adlandırılır). Bir alt program, bir kez verilen ve diğer birçok tarifin parçası olarak kullanılan bir sos tarifi gibidir. Alt programlar girdileri (gerekli miktar) alır ve sonuçlar üretir (sos). Yaygın olarak kullanılan alt programlar genellikle bir dille sağlanan bir koleksiyon veya kitaplarda bulunur. Alt programlar, aşağıdaki rutinde gösterildiği gibi tanımlarında diğer alt programları çağırabilir (burada ABS mutlak değil er fonksiyonudur). SQUARE_ROOT, iyi bir sonuç üreten bir WHILE (belirsiz) döngüsü kullanılarak uygulanır.

x çok küçük veya çok büyük olmadıkça gerçek sayıların karekökü için yaklaşım . Adını, girdi verisinin tipini ve ıktisini bildirerek bir alt program yazılır: FUNCTION SQUARE_ROOT(REAL x)
 RETURNS REAL ROOT 1.0 WHILE ABS(ROOT*ROOT - x) < 0.000001 AND WHILE ROOT = (x/
 ROOT + ROOT)/2 RETURN ROOT Alt programlar bir problemi daha küçük, daha izlenebilir alt problemlere bölebilir. Bazen bir problem, orijinalin daha küçük bir versiyonu olan bir alt probleme indirgenerek çözülebilir. Bu durumda rutin, özyinelemeli bir alt program olarak bilinir, çünkü sorunu tekrar tekrar kendisini çağırarak çözer. Örneğin, matematikteki faktöriyel fonksiyon ($n! = n \cdot (n-1) \cdot 2 \cdot 1$ —yani, ilk n tamsayının çarpımı), yinelemeli bir rutin olarak programlanabilir: FUNCTION FACTORIAL(INTEGER n) TAM SAYIYI DÖNDÜR EĞER n = 0 SONRA DÖNÜŞ 1 ELSE DÖNÜŞ n * FACTORIAL(n - 1)

Özyinelemenin avantajı, genellikle kesin bir tanımın basit bir yeniden ifade edilmesidir, yinelemeli bir çözümün muhasebe ayrıntılarından kaçınan bir tanımdır.

Makine dili düzeyinde, döngüler ve koşullar, programda yeni bir noktaya "atla" diyen dal talimatlarıyla uygulanır. Daha yüksek seviyeli dillerdeki "goto" ifadesi aynı işlemi ifade eder, ancak insanların bir programın "akışını" takip etmesini zorlaştırdığı için nadiren kullanılır. Java ve Ada gibi bazı diller buna izin vermez.

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Yazım farklılıkları; modern; Yönetim yapısı;
 operasyonel talimatlar; sıra; müdür
 varsayılan yapı; aritmetik işlemleri gerçekleştirmek; kök bulma; zincirleme; uygun tekrarlar;
 karar ver
 Sorunlar; içermek; koşullu döngü operatörü; varsayımlar;

hesaplama algoritması; mutlak değer er fonksiyonu; özyinelemeli alt program; kesin tanım.

1.2. Metinden aşağıdağı kelimelerin eş anlamlılarını bulun

1. Çağdaş, 2. kontrol, 3. yapı, 4. atamak, 5. bildirmek, 6. kök, 7. kesin, 8. takip etmek, 9. rutin, 10. tekrarlamak.

1) emir, 2) mevcut (yeni, geç), 3) yinelemek, 4) yapma biçimi, 5) vermek (değer er), 6) ilan etmek, 6) prosedür (işlem), 7) kök, 8) iyi tanımlanmış, 9) uygulama, 10) düzen.

1.3. Aşağıdağı kelimeleri tanımlarıyla eşleştiriniz

1. Koşullar	a) talimatların veya süreçlerin art arda yürütülmesini gerektiren bir işlem. b) bir programda bir görevi gerçekleştiriren bir dizi talimat.
2. Yinelemeli işlem	Programlar, "alt rutinler" ve sıkılıkla "fonksiyonlar" olarak da adlandırılan birçok rutinden oluşur. c) bir şekil, sembol veya formülle temsil edilmek. d) Bir şeyi tekrar tekrar yapma eylemi. e) İki miktar veya değerinin aynı olduğuunu söyleyen matematiksel bir ifade,
3. Döngü	örneğin $6 \times 4 = 12 \times 2$. f) bir koşul sağlanırken tekrar etmeye devam
4. Döngü 5. Rutin	eden bir döngü
6. denklem	doğru.
7. Ekspres	g) yalnızca belirli koşullar altında çalışan ifadeler.

1.4. Aşağıdağı soruları metne göre cevaplayınız

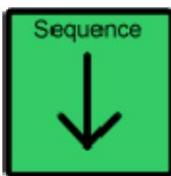
- 1) Hemen hemen her prosedür dilinde kaç temel kontrol yapısı vardır?
- 2) Bir dizi ne içerebilir?
- 3) SQUARE_ROOT nasıl uygulanır?
- 4) Özyinelemenin ana avantajı nedir?
- 5) Hangi tür programlama yapısı, bir eylemi atlama veya başka bir eyleme ayrılma olasılığı olmaksızın her komutun sırayla gerçekleştirilmesini gerektirir?

- 6) if/else ifadesi koşullu olarak iki ifadeyi dē̄lendirir. bu mu dō̄ḡ̄ ru?
- 7) Tekrarlanan işlemler iç̄̄n hangi kontrol yapısı kullanılır?
- 8) Tek bir mantıksal işlemi birlikte sağlayan ifadeleri gruplamak iç̄̄n hangi kontrol yapısı kullanılır?

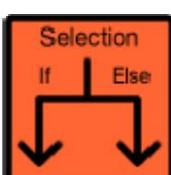
1.5. Aşağıdaki cümlelerde boşlukları doldurmak iç̄̄n dō̄ḡ̄ ru kelimeyi kullanın

tek bir mantıksal işlem	bir dizi ifade
dē̄ işkeni varsayılan olarak	dē̄ iştirmek iç̄̄n bir test
bir alt program	Süre

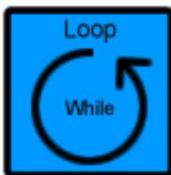
Program Akışı Kontrol



Sıra, yukarıdan aşağıya tek tek yürütülen a)'dan oluşur. Sıra, birçok programlama dili iç̄̄n b) kontrol akışıdır. Şimdiye kadar gösterilen tüm programlar, yürütülmeleri iç̄̄n bu kontrol akışını kullandı.



Seçim kullanılır c) iki veya daha fazla eylem arasında bir seçim yapılması gerektiğinde kontrol akışı. Genellikle seçim, programdaki bazı d)'lerin durumuna bağlıdır. Bu kontrol yapısı, genellikle If ve Else anahtar sözcükleri kullanılarak belirtilir.



Döngü, bir dizi ifadenin tekrar tekrar yürütülmesine neden olan bir kontrol yapısıdır. Her döngü yinelemesinde, döngünün devam edip etmeyeceğini belirlemek iç̄̄n e) gerekli şartlar ~~kefihi~~ ~~de~~ ~~bul~~ ~~kontrol~~ belirtilir.

_____ .



Alt programlar, ifadeleri gruplandırmanın bir yoludur g) _____ Önek bir alt program ~~üretilebilir~~ ~~ve karekökləmə~~ programa döndürebilen SquareRoot olabilir. Çağrı anahtar sözcüğü h) gösterir.

_____.

Görev 1.5'in anahtar

- cevapları. a) bir dizi ifade b)
- varsayılan c) değil istirmek
- için d) değil işkenler e) test
- etmek için f) süre g) tek bir
- mantıksal işlem h) bir alt
- program

2. Dilbilgisine Odaklanın

2.1. Mastarın İ şlevleri tablosunu inceleyin ve kendinizinkini oluşturun cümleler

Ders	<u>Bir sorunu daha küçük,</u> <u>daha izlenebilir alt</u> <u>programlara bölmektedir.</u>	Sorunu şuraya bölün daha küçük, daha anlaşılabilir rutinler zor değil il.
Zarf amaç değil istircisi (sırayla vb . tanıtılabılır)	Bilgisayarın belleğ ini, 'a', '?' veya 'Z' gibi harfler ve karakterler gibi <u>diğer veri türlerini depolamak için de kullanabiliriz.</u>	biz de yapabiliriz kullanmak hafıza diğer depolama işin bilgisayarı türleri harfler gibi veriler ve benzeri semboller «ah» , «?» veya «Z» .
Sonucun zarf değil istircisi (esas olarak zarflar tarafından yeterince ve çok değil istirilen sıfatlardan sonra ve as birleşiminden sonra ortaya çıkar)	Buluntular hakkında <u>konuşulamayacak kadar az.</u> Kural, herkes tarafından kolayca <u>gözlemlenebilecek şekilde formüle edilmiştir.</u>	Onlar hakkında söylenecek çok az bulgu var (idi) konuşmak. Kural, herkesin kolayca anlayabileceğ i şekilde formüle edildi. onu gözlemleyin.
tahmin edici	Bir değil işkenin türünü belirtmenin tipik bir yolu, tür adını değil işken tanımlayıcıdan önce yazmaktır.	tipik bir şekilde türünü belirtin değ işken önce tür adını yazma tanımlayıcı değ işken. Bu
Bağlanmak	Bu, dikkate alınması gereken ana <u>avantajdır.</u>	ana avantajı dikkate alınması gereklidir.
Nesne	Koşullu IF-THEN veya IF-THEN-ELSE kontrol yapısı, bir programın alternatif <u>yürütmeye yollarını izlemesine izin verir.</u>	koşullu kontrol IF-THEN yapısı veya IF-THEN-ELSE programa izin verir alternatif yolları takip et uygulamak.

2.2. Mastarların biçim ve işlevleri hakkında yorum yapın ve aşağıda cumleleri çevirin

- 1) Girdi, her program tarafından kullanılan iki öğeden biridir.
çünkü her programın çalışması için bazı verilere ihtiyacı vardır.
- 2) Banka, denemeyen kişinin siz olmadığından emin olmak istiyor
hesabınıza erişmek için
- 3) Bilgisayarlar, bir para çekme veya yatırma işleminden sonra ek hesabı bakiyenizi
güncellemek için gereken basit toplama veya çıkarma işlemlerinden, bir uydunu
yörüngeye oturtmak için gereken karmaşık hesaplamaya kadar her türlü
matematiksel işlemi ve işlevi gerçekleştirebilir.
- 4) Bu elemanları kullanmak için ithal edilir.
- 5) Mevcut olanlara yeni isimler vermek için atama ifadelerini kullanabiliriz.
fonksiyonlar.
- 6) Bu bölümdeki hedeflerimizden biri, düşünme ile ilgili konuları izole etmektir.
prosedürel olarak.
- 7) Amacım, aşağıda idakiler gibi insanlarla etkileşime girebilecek bir program oluşturmaktı.
günümüzün sohbet robotları.
- 8) Terminalde çıktıyı görüntülemek için 'echo' komutu ve ardından görüntülenecek
metin kullanılır.

3. Tartışma

3.1. Aşağıda idakiler soruları partnerinizle tartışın

- 1) Programlar nasıl çalışır ve bunları nasıl oluşturabilirsiniz?
- 2) Algoritmalar kaç tane önemli kontrol yapısı gerektirir? Ne
onlar mı?

3.2. Sunumlar ve Raporlar için önerilen konular

- 1) Çeşitli Programlama Dillerinde SemiColon'un rolü 2) C++'daki yapılar 3)
İki yönlü seçim ve çok yönlü seçim kontrol yapıları.

4. Ek Okuma

4.1. Bazı bildirim dilleriyle ilgili aşağıdaki idaki metni okuyun ve çevirin

bildirim dilleri

Prosedürel olmayan veya çok yüksek seviye olarak da adlandırılan bildirim dilleri, bir programın (ideal olarak) nasıl yapılacağıını değil il ne yapılacağıını belirttiğ i programlama dilleridir. Bu tür dillerde, bir programın belirtilmesi ile uygulanması arasında, şimdiden kadar açıklanan prosedür dillerinden daha az fark vardır. İki yaygın bildirim dili türü mantık ve işlevsel dillerdir.

PROLOG'un (mantıkta programlama) en iyi bilinen olduğu mantık programlama dilleri, bir programı bir dizi mantıksal ilişki olarak belirtir (örneğin, büyük ebeveyn, birinin ebeveyninin ebeveynidir). Bu tür diller SQL veritabanı diline benzer. Bir program, bir soruyu yanıtlayacak çıkarımlar yapmak için bu ilişkileri sistematik olarak arayarak bir soruyu yanıtlayan bir "çıkarm motoru" tarafından yürütülür. PROLOG, doğal dil işleme ve diğer AI programlarında yaygın olarak kullanılmaktadır.

İşlevsel dillerin matematiksel bir stilini vardır. Fonksiyonların argümanlarına uygulanmasıyla fonksiyonel bir program oluşturulur. LISP, ML ve Haskell gibi fonksiyonel diller, dil geliştirmede, otomatik matematiksel teorem kanıtlayıcılarında ve bazı ticari projelerde araçları olarak kullanılır.

Komut dosyası dillerine bazen küçük diller denir. Büyük programları yönetilebilir hale getirmek için gereken veri bildirimlerinin ve diğer özelliklerin ek yükünü gerektirmeyen nispeten küçük programlama problemlerini çözmeyi amaçlarlar. Komut dosyası dilleri, işletim sistemi yardımcı programlarını yazmak, özel amaçlı dosya işleme programları ve öğrenmesi kolay oldukları için bazen çok daha büyük programlar için kullanılır.

Perl, 1980'lerin sonlarında, başlangıçta UNIX işletim sistemiyle kullanılmak üzere geliştirildi. Daha önceki komut dosyası dillerinin tüm özelliklerine sahip olması amaçlanmıştır. Perl, ortak işlemleri belirtmek için birçok yol sağladı ve böylece bir programcının herhangi bir uygun stili benimsemesine izin verdi. 1990'larda hem küçük yardımcı programlar hem de daha büyüklerin prototipleri için bir sistem programlama aracı olarak popüler hale

Aşağıda tartışılan diğer dillerin yanı sıra, bilgisayar Web "sunucularını" programlamak için de popüler hale geldi. (2200) [<https://www.britannica.com/technology/computer-programming-language/Visual-Basic>]

4.2. Sınıfta bazı bildirim dillerini tanımlayın

ÜNİ TE 8. Elementlerin programlama. Veri St ateskes bu

Öğrenmek nesnesi

derinliği temel c ve adv vanced ortak veri st yapı
kavramlarına dikkate r farklı veri st yapıları
nt türlerine o

anahtar kılardar ve d ifadeler s. Rusça ver eşittir ödünc verme hataları not
anlam anahtar kelimenin gols ve ph bize laf atıyoed içinde metin

Ty'dnin türleri; belirtmek için; ; saklamak parça; tam sayılar s; gerçek ekşiyalar; ;
haykarakter vektörler; dizi; bir ts toplanmış kayıtlı ;
ile birlikte bilesen veya alan ds; Özette m; dinamik mikrofon tahşit katyon; aintree; Öz t
Hangi türler s; merhaba de; uygulanır uygun; ; om'a i le birlikte; bilişmiş gizleme; ;
reü kullanılabılırlik; Öz tion; m'ye pu yapmak kalka açık tuval backup operüretim; i sokma n
operasyon.

okumadan önce videoyu izletşuradan tmetin

https://www.senTube.com/izle?fv=DuDz_z6B4cqV_Vc&ab_ch=ticaret=C_CrashC_ayi.R_Render_ts_icler_Rusça_nt_Asyas_açikla_nasilsın_anlamadıdayan_kavram_egitim

Teknik oku takip eden metin ve eski yapmak egzersizler arkadan verildi sonra

Wburada trol organize yapı idrar
e algoritmalarıms, veri yapısı ures
düzenlemek bilgilendirme. ben parçalar halinde küçük, d veri
yapısı spesifik türler veri bir, ve th hus
hangi o işlemleri s gerçek ekleştirilebilir üzerinde em,
kaldırılırken ting th ihtiyacı var d takip a
program boşver etmek için o notun yada
adres evet. Kılıbıkple veri res dani
tam sayılar, tam olarak de number, B Boole'ler s (doğru/false) ve d karakter ters veya karakter r
Teller. bileşik ve veriler içindir tarafından rm ve kombin tek veri yada daha fazla
tipi ning bu.

11001
 10001 11100110
 0010 110001 11000110
 00101001 01011010 1100
 010101 1100000100 100
 00011111 10 1001110
 00101 11010 10
 10010 101
 00100
 01001
 00110
 0000110

Data Structure

En önemli bileşik veri yapıları dizi, homojen bir koleksiyon ve kayıt, heterojen bir koleksiyondur. Bir dizi, bir sayı vektörünü, ~~bir dizi liste~~ ~~koleksiyonu~~ ~~ve kayıt~~ matematiksel matris) temsil edebilir. Bir kayıt, çalışan bilgilerini (isim, unvan ve maaş) depolayabilir. Çalışanlar tablosu gibi bir dizi kayıt, her biri heterojen olan bir ögeler topluluğudur. Tersine, bir kayıt bir vektör, yani bir dizi içerebilir.

Kayıt bileşenleri veya alanları ada göre seçılır; örneğin, E.SALARY, E kaydının maaş alanını temsil edebilir. Bir dizi ögeli, konumu veya indeksi tarafından seçılır; A[10], A dizisindeki 10. konumdaki ögeli edir. Bir FOR döngüsü (belirli yineleme), ögelerini toplamak için dizin sınırlarına sahip bir dizide (aşağıdaki örnekte BİRİNÇİ DEN SONRA) çalışabilir: FOR i FIRST SON TOPLA TOPLA + A[i]

Diziler ve kayıtlar sabit boyutlara sahiptir. Büyüyebilen yapılar, gerektiğiinde yeni depolama sağlayan dinamik tahsis ile inşa edilir.

Bu veri yapıları, her biri veri ve diğer bileşenlere referanslar (makine terimleriyle, adresleri) içeren bileşenlere sahiptir. Bu tür kendi kendine referanslı yapıların özyinelemeli tanımları vardır. Örneğin bir bintree (ikili ağ aç) ya boştur ya da verileri ve sol ve sağ bintree "alt ögeleri" içeren bir kök bileşeni içerir. Bu tür bintree'ler, bilgi tablolarını verimli bir şekilde uygular. Bunlar üzerinde çalışacak alt programlar doğrudan olarak özyinelemelidir; aşağıdaki yordam bir bintree'nin tüm ögelerini yazdırır (her biri bazı alt ağlarının köküdür): PROSEDÜR TRANSFER(ROOT: BINTREE)

```

EĞER DEĞİL(BOŞ(KÖK))
HAREKET(KÖK.SOL)
KÖK.VERİ GEÇİŞİ
(KÖK.SAĞ YAZDIR
ENDIF

```

Özet veri türleri (ADT'ler), büyük ölçüde ekli programlama için önemlidir. Veri yapılarını ve üzerlerindeki işlemleri paketleyerek dahili ayrıntıları gizlerler. Örneğin, bir ADT tablosu, bir dizi, liste veya ikili ağ olsun, temel yapıyı görünmez tutarken kullanıcılara eklemeye ve arama işlemleri sağlar. Nesne yönelik dillerde sınıflar ADT'lerdir ve nesneler de bunların örnekleridir. Aşağıdaki nesne yönelik sınıflar

sözde kodörneği, bir karşılaştırma işleminin (tamsayılar için "<" gibi) olduğunu verileri karakterize eden bir ADT bintree ve bir "üst sınıf" KARŞILAŞTIRILABİ Lİ R olduğunu varsayar. Veri temsilini gizleyen ve tablolara uygun işlemleri sağlayan yeni bir ADT, TABLE'ı tanımlar. Bu sınıf polimorfiktir; COMPARABLE sınıfının eleman tipi parametresi olarak tanımlanır. Bunun herhangi bir örneği, bu türü, burada çalışan verilerini içeren bir sınıfı belirtmelidir (KARŞILAŞTIRILABİ Lİ R bildirimi, PERS_REC'in kayıtları sıralamak için bir karşılaştırma işlemi sağlama gereği anlamına gelir). Uygulama ayrıntıları atlanmıştır.

<KARŞILAŞTIRILABİ Lİ R T> SINIF TABLOSU

ÖZEL VERİ : <T> BINTREE

GENEL EK (ÖE: T)

GENEL ARAŞTIRMA(ITEM: T) BOOLE GERİ DÖNDÜ
SON

SINIF PERS_REC: KARŞILAŞTIRILABİ Lİ R
ÖZEL ADI: STRING

ÖZEL POZİ SYON: {PERSONEL, MÜDÜR, MÜDÜR}
ÖZEL MAAŞ: GERÇEK

KAMU KARŞILAŞTIRMA (R: PERS_REC) BOOLE GERİ DÖNDÜ
SON

ÇALIŞANLAR: TABLO <PERS_REC>

TABLE, yalnızca kendi işlemlerini kamuya açıklar; bu nedenle, bintree yerine bir dizi veya liste kullanacak şekilde değil istirilirse, onu kullanan programlar değil işiklığı algılayamaz. Bu bilgi gizleme, büyük programlarda karmaşaklığını yönetmek için gereklidir. Parçalar arasında "sözleşmeler" ile onları küçük parçalara böler; burada TABLE sınıfı, arama ve ekleme işlemleri sağlamak için sözleşme yapar ve kullanıcıları yalnızca bu şekilde duyurulan işlemleri kullanmak için sözleşme yapar.

Veri yapılarının avantajları Veri yapısının avantajları şunlardır: Verimlilik: Belirli bir ADT'yi uygulamak için bir veri yapısı seçimi uygunsa, programı zaman ve mekan açısından çok verimli hale getirir.

Yeniden Kullanılabilirlik: Veri yapısı yeniden kullanılabilirlik sağlar, yanı

birden çok istemci programı veri yapısını kullanabilir.

Soyutlama: Bir ADT tarafından belirtilen veri yapısı aynı zamanda soyutlama düzeyini de sağlar. İstemci, veri yapısının dahili çalışmasını göremez, bu nedenle uygulama kısmı hakkında endişelenmesi gerekmek. İstemci yalnızca arayüzü görebilir. [<https://www.javatpoint.com/data-structure-tutorial>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Tasarım algoritmaları; özellikle; tüm sayılar; boole
değerler; bileşik veri yapıları; homojen toplama
veri; kayıt; tersine; takdim etmek; diziler; indeks; belirli yineleme; dinamik bellek
tahsis; ikili (ikili) ağ aç; örnekler; verilerin sunumu; değer işiklikleri algılamak; sıralı
arama; olasılık

yeniden kullanım; soyut veri türü (benzersiz
nesnelere uygulanan terimlerle tanımlanan veri türü
işlemleri (yani, bir dizi erişim işlevi)).

1.2. Aşağıdaki kelimeleri tanımlarıyla eşleştirin:

1. Ağ aç topolojisi	a) birlikte grüplendirilmiş bir grup ilgili veri değerleri (eleman olarak adlandırılır). Aynı veri türü olmalıdır. b) bir işaretçi veya yalnızca tek boyutlu bir dizi. Bilgisayar
2. Sözde kod	grafiklerinde bu terim, başlangıç ve bitiş noktası olan bir çizgiyi tanımlar. c) Birbirine bağlı birçok elemanın bir ağ açın dalları gibi düzenlendiği özel bir yapı tipi.
3. Dizi	Örneğin, bir şirket ağındaki bilgisayarları veya bir veritabanındaki bilgileri düzenlemek için sıkılıkla kullanırlar. d) Nihai sonuca ulaşmak için kendisinin daha küçük bir parçasını tekrar tekrar hesaplayan bir işlev veya yöntem. Yinelemeye benzer.
4. Alan	

5. Veri yapısı	e) Düz İngilizcaye benzeyen, derlenemeyen veya yürütülemeyen, ancak bir sorunun çözümünü açıklayan bir bilgisayar programlama dili. f) bir veri tabanı veya
6. Özyinelemeli	yazılım programı içindeki bir sütunda yer alan tek bir veri öğesi. Örneğin, bir müşteri adı, adresi veya telefon numarası olabilir. g) bir bilgisayar programında verileri
7. Vektör	verimli bir şekilde depolamak, bunlara erişmek ve işlemek için önceden tanımlanmış bir format.

1.3. Aşağıdaki soruları metne göre cevaplayınız

- 1) Veri yapıları bilgisayar programlama açısından ne yapar?
- 2) Basit veri yapıları neleri içerir?
- 3) Bir dizi elemanı nasıl seçilir?
- 4) Bir ADT tablosu ne sağlar?
- 5) Ağ aç veri yapısı nedir?
- 6) Aşağıdaki veri yapılarından hangisi (diziler, kayıtlar, işaretçiler) homojen olmayan veri öğelerini depolayamaz?

1.4. Metni tekrar okuyun ve aşağıdaki ifadelerin doğrusu mu yoksa doğrusu olmadığına karar verin.
yanlış.

- 1) Veri yapıları, bir programcının bellek adreslerini takip etme ihtiyacını ortadan kaldırarak, veri türlerini ve böylece bunlar üzerinde hangi işlemlerin gerçekleştirilebileceğiini belirtir.
- 2) Bileşik veri yapıları, bir veya daha fazlasının birleştirilmesiyle oluşturulur. veri tipleri.
- 3) En önemli bileşik veri yapıları dizi, of ve kayıttır, homojen bir koleksiyon veri, heterojen bir koleksiyon.
- 4) Çalışanlar tablosu gibi bir dizi kayıt, her biri homojen olan elementlerdir.
- 5) Bir dizi elemanı konumu veya indeksi ile seçilir; A[10], A dizisindeki 10 konumundaki öğedir.
- 6) Dizi ve kayıtların sabit boyutları yoktur. Büyüyebilen yapılar, gerektiğiinde yeni depolama sağlayan dinamik tahsis ile inşa edilir.

- 7) Soyut veri türleri, büyük ölçüklü programlama için önemlidir. Veri yapılarını ve üzerindeki işlemleri paketleyerek dahili ayrıntıları gizlerler.
- 8) Örneğin, bir ADT tablosu, bir dizi, liste veya ikili ağ aç olsun, temel yapıyı görünmez tutarken kullanıcılar ekleme ve arama işlemleri sağlar.
- 9) Bir ADT tarafından belirtilen veri yapısı, gereklili soyutlama seviyesini sağlanamaz. İstemci, veri yapısının dahili çalışmasını göremez.
- 10) İstemci sadece arayüzü görebilir.

Görevin anahtar cevapları 1.2.

1.c); 2.e); 3 A); 4.f); 5.g); 6.d); 7.b).

2. Dilbilgisine Odaklanın

2.1. Parantez içindeki fiillerin mastar veya mastar hallerini kullanarak cümleleri tamamlayınız.

- 1) Beni gördü (kapattı) bilgisayarı.
- 2) Patronumuz genellikle tüm personeli tazeleme kurslarına teşvik eder (alır).
- 3) Ona tüm makbuzları bir kez daha yaptı (kontrol edin).
- 4) Sunumumu kesmeye cesaret edemediler.
- 5) Programlama derslerine karar verdi (devam etti).
- 6) Seni böyle (düşündüren) ne?
- 7) Tasarım projemi yöneticiye tavsiye etti (gösterdi).
- 8) Ofisten hemen ayrılmamı emrettiler.
- 9) Bu şirketi en güvenilir ortak olarak görüyoruz.
- 10) Verilen tüm bilgileri bana (analiz etme) yardımcı oldu.
- 11) Müzakereleri duydum (dur).
- 12) Ailem orada geç kalmama (kalmama) izin vermedi.
- 13) Postayı akşam sekreterimize (kontrol edin) söyledik.
- 14) Onları (konuşmalarını) yüksek sesle işittim.
- 15) Bazı bilim adamları, Mars'ı (örtülü) bitki örtüsü ile düşünürler.
- 16) Bir başka olasılık da kuvars (kullanmak) idi.

Mastar Yapılar

Karmaşık Nesne (Mastar yapımı ile Amaç)

Ortak durumda	Öne Yüklem İ sim	+ mastar
	Nesnel durumda zamir	
Bu testin sonuçları	ha inandi	çizildi/şemada.

Öneğ in , koşullar onu şehri terk etmeye zorladı. -

Koşullar onu şehri terk etmeye zorladı (zorladı).

NOT

Görmek, fark etmek fiillerinden sonra duyu algısını ifade ettiklerinde "olmak" fiilinin mastarı kullanılmaz. Bunun yerine bir yan cümleciği kullanılır. Öneğ in , iç eride olduğu unu gördük. – Я увидел, что он дома.

Karmaşık Öne (Özel Mastar Yapısı)

Ortak durumda isim veya aday durumda zamir	Sonlu bir fiil	Sonsuz	
Ressam			
O	görünüyordu	Görmek	hiç bir
antlaşma	söylendi	şey. dün olmak. imza	ı

Mastar için yapı

	edat	İ sim zamir	sonsuz
Onlar bekledi	için	kapı	açmak.
İ şe yaramaz	için	ben	onunla konuşmak için.

Öneğ in Bu soruya cevap vermek benim için kolaydı. -

Bu soruya cevap vermek benim için kolaydı.

2.2. Aşağı ıda verilen cümleleri ilgili sütunlara göre sıralayınız. İ lk ikisi sizin için yapılır.
Onları Çevir

karmaşık Nesne	1,
Karmaşık Konu.	2,

- 1) Dö nüş, vedayı sevdirir.
- 2) Onun politikası politika değil ildir. Ve bunun kendi içinde bir politika olması gerekiyordu.
- 3) Yapamayacaklarınızın yapabileceklerinizi engellemesine izin vermeyin.
- 4) Geleneksel toast 'Aşağı idan yukarıya' kesinlikle tabu olarak bilinir.
Donanma.
- 5) Son damla bardağın taşmasına neden olur.
- 6) Büyükelçilerin devletlerin gözü kulağı olduğu u söylenir.
- 7) Diplomasi, sıkıntılı sulara atlama sanatı olarak düşünülür.
sıkrama yapmadan.
- 8) Deneyimin, biz olduğumuzda doğanın bize verdiği bir tarak olduğum u söylenir.
gözü pek.
- 9) Çok u zaman işlerin olduğundan farklı olduğum ortaya çıkar.
ilk olarak görülmektedir.
- 10) Atı suya götürülebilirsin ama ona içiremezsin.
- 11) Kararın yarın kabul edileceği i kesin ve
onunla tanışmış.
- 12) Bununla birlikte, Wilkinson'ın işaret ettiği gibi, hızlı geçişlerin yavaş olanlardan daha muhtemel olduğum doğrudur.
- 13) Mühendis, işçilerin elektrik için yumuşak kauçuk kullanımını istiyor.
yalıtım.

3. Tartışma

3.1. 'Kontrol Yapısı' ve 'Veri Yapısı' terimlerinin kısa bir açıklamasını yapın

3.2 Aşağıdaki soruları sınıfta tartışın. Gerekirse bazı ek İnternet kaynaklarını kullanabilirsiniz.

- 1) Veri Yapısı ve algoritmalar için en iyi dil hangisidir? Çok u rekabetçi programcı C++ kullanır. Neden böyle olduğumu açıklayabilir misin?
- 2) DSA öğrenmeye nasıl başlarsınız?
- 3) Veri yapısını ve algoritmaları öğrenmek zor mu?
- 4) Python veri yapıları nelerdir?
- 5) Veri yapısını ve algoritmaları öğrenmek kaç gün sürer?

3.3. Çeşitli programlama dillerini kullanmanın gelecekteki bekleyenleri hakkında bir rapor hazırlayın

S.No.	Diller	Gelecek Kapsamı
1.	python	Python'un şüphesiz programlama dili geliştirme alanında, özellikle veri gorselleştirme, yapay zeka, veri bilimi ve makine öğrenimi disiplinlerinde parlak bir geleceği var.
2.	Java	Java birçok işletmede yaygın olarak kullanılmaktadır. Ayrıca çeşitli mallar yapmak için kullanılabilir ve çok çeşitli kullanımlara sahiptir. Şu anda en yaygın kullanılan programlama dilidir, bu yüzden öğrenmeye oldukça değer er.
3.	C++	C++ çok çeşitli uygulamalara sahiptir ve onu incelemek asla kötü bir şey değil. Almak ve anlamak için çok basit bir dildir. Endüstride geniş bir uygulama alanına sahiptir. Grafik tasarımlar ve 3 boyutlu modellerin yanı sıra oyunlarda da kullanılmaktadır.
4.	C	C bazı uygulamalarda güncelliği ini yitirmiş olsa da, yakın zamanda ortadan kalkmıyor. Çok çeşitli gerçek dünya uygulamalarına sahiptir ve endüstride uzun yıllar kullanılmaya devam edecektir.
5.	C#	C#, her ikisi de oyun endüstrisine faydalayan oyuncuların üretmedeki etkili yetenekleri ve dayanıklılığı nedeniyle popülerlik kazanan ve önumzdeki yıllarda da öyle kalması muhtemel bir dildir. Ayrıca iş uygulamalarında oldukça faydalıdır.
6.	Javascript	JavaScript yaygın olarak kullanılan bir programlama dilidir. O kadar yaygın olarak kullanılıyor ki, başka bir programlama dilinin yerini alması uzun zaman alabilir. Web geliştirmenin yanı sıra yapay zeka ve diğer alanlarda da kullanılmaktadır.

		Bu dil, herkesin ög renme önceliğ i listesinin başında olmalıdır.
7.	yakut	Günümüz dünyasında, Ruby hala çok sayıda uygulama için kullanılmaktadır. Sonuç olarak, ög renmek için harika bir dil çünkü kısa sürede karmaşık uygulamalar oluşturabileceksiniz. Aynı zamanda sağ lam bir teknolojiye sahiptir. Bu nedenle bugün hala geçerlidir.

4. Ek Okuma

4.1. Aşağı idaki metni okuyun ve çevirin

C# (/si ſa:rp/ see sharp) genel amaçlı, çok paradigmalı bir programlama dilidir. C#, statik yazma, güçlü yazma, sözlüksel olarak başa çıkma, zorunlu, bildirimsel, işlevsel, genel, nesne yönelimi (sınıf tabanlı) ve bileşen yönelimi programlama disiplinlerini kapsar.

.NET Framework'ün geliştirilmesi sırasında, sınıf kitaplıkları orijinal olarak "Simple Managed C" (SMC) adı verilen bir yön netilen kod derleyici sistemi kullanılarak yazılmıştır.

Ocak 1999'da Anders Hejlsberg, "C-like Object Oriented Language" anlamına gelen Cool adında yeni bir dil oluşturmak için bir ekip kurdu. Microsoft, "Cool" adını dilin son adı olarak tutmayı düşünmüştü, ancak ticari marka nedenleriyle bunu yapmamayı seçti. .NET projesi Temmuz 2000 Profesyonel Geliştiriciler Konferansı'nda herkese açık olarak duyurulduğunda, dil C# olarak yeniden adlandırıldı ve sınıf kitaplıkları ve ASP.NET çalışma zamanı C#'a taşındı.

Hejlsberg, C#'ın Microsoft'ta baş tasarımcısı ve baş mimarıdır ve daha önce Turbo Pascal, Embarcadero Delphi (eski adıyla CodeGear Delphi, Inprise Delphi ve Borland Delphi) ve Visual J++ tasarımında yer almıştır. Röportajlarda ve teknik makalelerde, çok u büyük programlama dilindeki (örneğ in C++, Java, Delphi ve Smalltalk) kusurların Ortak Dil Çalışma Zamanının (CLR) temellerini ve bunun da C# dilinin tasarımını yönlendirdiği ini belirtti. kendisi.

1994 yılında Java programlama dilini yaratan James Gosling ve Java'nın yaratıcısı Sun Microsystems'in kurucularından Bill Joy, C#'ı Java'nın "taklidi" olarak adlandırdı; Gosling ayrıca "[C#]" bir tür

Güvenilirlik, üretkenlik ve güvenlik ile Java silindi." Klaus Kreft ve Angelika Langer (bir C++ akış kitabı'nın yazarları) bir blog yazısında "Java ve C# neredeyse aynı programlama dilleridir. İlk novasyondan yoksun sıkıcı tekrarlar", "Java veya C#'ın program yazma şeklimizi değil istiren devrim niteliğindeki programlama dilleri olduğumu pek kimse iddia edemez" ve "C# Java'dan çok şey ödünç aldı - ve tam tersi. Artık C#, boks ve kutudan çıkarmayı desteklediğiine göre, Java'da çok benzer bir özelliğe sahip olacağınız." Temmuz 2000'de Hejlsberg, C#'ın "Java klonu olmadığıını" ve tasarımda "C++'a çok daha yakın" olduğunu söyledi.

C# 2.0'ın Kasım 2005'te piyasaya sürülmüşinden bu yana, C# ve Java dilleri giderek farklılaşan yörüneler üzerinde gelişti ve oldukça farklı iki dil haline geldi. İlk büyük sapmalardan biri, her iki dile çok farklı uygulamalarla jeneriklerin eklenmesiyle geldi. C#, sınıf yükleme zamanında gerçekleştirilen kod oluşturma ile diğer herhangi bir sınıf gibi kullanılabilen "birinci sınıf" genel nesneler sağlama için şeyleştirmeyi kullanır.

Ayrıca, C#, işlevsel stil programlamayı barındırmak için birkaç ana özellik ekledi ve C# 3.0 ile yayınlanan LINQ uzantıları ve onun destekleyici lambda ifadeleri, uzantı yöntemleri ve anonim türler çerçevesiyle sonuçlandı. Bu özellikler, C# programclarının, uygulamaları açısından avantajlı olduğunu unda, kapatmalar gibi işlevsel programlama tekniklerini kullanmalarını sağlar. LINQ uzantıları ve işlevsel içe aktarmalar, geliştiricilerin bir veritabanını sorgulamak, bir XML dosyasını ayırtmak veya bir veri yapısında arama yapmak gibi yaygın görevlere dahil edilen standart kod miktarını azaltarak, vurguyu gerçek program mantığına kaydırarak okunabilirliği artırırmaya yardımcı olur. ve sürdürülebilirlik (2800)

[[https://en.wikipedia.org/wiki/C_Sharp_\(programming_language\)](https://en.wikipedia.org/wiki/C_Sharp_(programming_language))]

4.2. Cevap için uygun bir soru sorun

- 1) C# , genel amaçlı, çok paradigmalı bir programlama dilidir.
- 2) Hejlsberg, Microsoft'ta C#'ın baş tasarımcısı ve baş mimarıdır.
- 3) Ocak 1999'da.
- 4) Bu özellikler, C# programclarının, uygulamaları için avantajlı olduğunu unda, kapatmalar gibi işlevsel programlama tekniklerini kullanmalarını sağlar.
- 5) LINQ uzantıları ve işlevsel içe aktarmalar.

4.3. Aşağıdaki cümleleri çevirin

- 1) Bir dizi, aşağıda idakilerden oluşan yapılandırılmış bir veri türüdür: aynı türden sabit sayıda eleman, birleştirilmiş her öğenin kendi numarasına (dizin) sahip olduğu u bir ad.
- 2) Daha önce de belirttiğimiz gibi, algoritmalar iki önemli kontrol yapıları: yinelemeler ve seçim içindir. 3) Her ikisi de Python'da çeşitli biçimlerde desteklenir. Programcılar daha fazla olacak yöntemi seçebilir durumlarda uygundur.
- 4) Yinelemeler için Python, standart while ifadesini sunar ve ifade için çok güçlü. 5) Select ifadeleri programcının soru sormasına izin verir ve cevaba göre çeşitli eylemler gerçekleştirir. 6) Çoğu u programlama dili iki faydalı yapıların sürümleri: ifelse ve if. Basit örnek ifelse ifadesinin ikili kullanımı:

`n<0 ise :`

```
print("Maalesef değer er negatif")
```

`başka:`

```
yazdır(matematik.sqrt(n))
```

- 7) Perl'in sözdiziminin çoğu u C diline dayandığıından, C dillerini bilen programcılar için, C++, C#, Java, JavaScript, Python veya PHP, Perl sözdizimi çok tanındık.
- 8) Bazı durumlarda (örneğin, dosya özniteliklerini Unix) sekizli sistemde sayıları daha açık bir şekilde temsil eder hesaplaşma.
- 9) Dizeler ve sayılar arasındaki dönüşümlerin yapılması çok uygundur. bağımlı olarak otomatik olarak yürütülür kullandıkları ifade. 10) Perl'de dil yapılarının anlamını netleştirmek için genellikle bağımlı kavramını kullandı, bu da kullanımını belirleyen bir dil öğesinin (değer işken, alt program vb.) programlama ortamı.

ÜNİ TE 9. Web – Geliştirme. Web Türleri – geliştirme

Öğrenme hedefleri

Web geliştirme ve türleri hakkında temel bilgileri edinmek Web geliştirme
ile web arasındaki farkı düşünmek
tasarım

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

intranet; işlevsellik; iç erik yönetim sistemleri (CMS); düz metin; Web içeriği geliştirme; Müşteri irtibat bürosu; ağ güvenliği yapılandırması; Çevik Metodolojiler; ön uç geliştirici; arka uç geliştirici; tam yığın geliştirici; veritabanı teknolojisi; Yerleşim; yazı tipleri; sorunsuz çalıştırmak için; Aşağı İya doğru açılan menü; kaydırma çubukları; ödemeye işlevi; Kullanıcı deneyimi tasarım; Kullanıcı arayüzü tasarım; görsel tasarım; parçalara ayrılacak; ilişkisel veritabanı yönetim sistemi (RDBMS).

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.



Web geliştirme, internet veya intranet olarak bilinen özel bir ağ için web siteleri ve uygulamalar oluşturma sürecidir.

Web geliştirme, bir web sitesinin tasarımlıyla ilgili değil; bunun yerine, tamamen web sitesinin işlevselliliğini sağlayan kodlama ve programlama ile ilgilidir.

En basit, statik web sayfalarından sosyal medya platformları ve uygulamalarına, e-ticaret sitelerinden iç erik yönetim sistemlerine

(CMS) - İnternet üzerinden günlük olarak kullandığımız tüm araçlar geliştiriciler tarafından oluşturulmuştur.

Web geliştirme, basit bir tek statik düz metin sayfası geliştirmekten karmaşık web uygulamalarına, elektronik işletmelere ve sosyal ağ hizmetlerine kadar değişebilir. Web geliştirmenin yaygın olarak atıfta bulunduğu daha kapsamlı bir görev listesi, Web mühendisliği, Web tasarıımı, Web içeriği geliştirme, istemci bağlantıları, istemci tarafı/sunucu tarafı komut dosyası oluşturma, Web sunucusu ve ağ güvenliği yapılandırması ve e-ticaret geliştirmeyi içerebilir.

Daha büyük organizasyonlar ve işletmeler için, Web geliştirme ekipleri yüzlerce kişiden (Web geliştiricileri) oluşabilir ve Web siteleri geliştirirken Çevik metodolojiler gibi standart yöntemleri takip edebilir. Web geliştirme, belirlenmiş bir departmanın etki alanından ziyade departmanlar arasında ortak bir çaba olabilir.

Üç tür Web geliştirici uzmanlığı vardır: ön uç geliştirici, arka uç geliştirici ve tam yapı in geliştirici. Ön uç geliştiriciler, kullanıcı tarayıcısında çalışan davranış ve görsellerden sorumluyken, arka uç geliştiriciler sunucularla ilgilenir.

Web geliştirme türleri Web

Geliştirme üç katmana ayrılabilir: istemci tarafı kodlama (ön uç), sunucu tarafı kodlama (arka uç) ve veritabanı teknolojisi.

İstemci

Tarafı İstemci tarafı komut dosyası oluşturma veya ön uç geliştirme, son kullanıcının doğrudan deneyimlediği her şeyi ifade eder. İstemci tarafı kodu, bir web tarayıcısında yürütülür ve insanların bir web sitesini ziyaret ettiklerinde gördükleriyle doğrudan ilgilidir. Düzen, yazı tipleri, renkler, menüler ve iletişim formları gibi şeylerin tümü ön uç tarafından yönlendirilir.

Sunucu tarafı

Sunucu tarafı komut dosyası oluşturma veya arka uç geliştirme, perde arkasında neler olup bittiğiyle ilgilidir. Arka uç, aslında bir web sitesinin kullanıcının gerçek ekten görmediği kısmıdır. Verileri depolamaktan ve düzenlemekten ve istemci tarafında her şeyin sorunsuz çalışmasını sağlamak sorumludur. Bunu ön uç ile iletişim kurarak yapar. İstemci tarafında bir şey olduğunda - örneğin, bir kullanıcı bir form doldurur - tarayıcı sunucu tarafına bir istek gönderir. Sunucu tarafı, tarayıcının daha sonra yorumlayabileceği ve görüntüleyebileceği ön uç kodu biçiminde ilgili bilgilerle "yanıt verir".

Veritabanı teknolojisi Web

siteleri de veritabanı teknolojisine güvenir. Veritabanı, bir web sitesinin çalışması için gerekli olan tüm dosyaları ve içeriği iç erir ve onu almayı, düzenlemeyi, düzenlemeyi ve kaydetmeyi kolaylaştıracak şekilde saklar. Veritabanı bir sunucu üzerinde çalışır ve çoğu web sitesi tipik olarak bir tür ilişkisel veritabanı yönetim sistemi (RDBMS) kullanır.

Özetlemek gerekirse: öncü, arka uç ve veritabanı teknolojisi, tamamen işlevsel bir web sitesi veya uygulama oluşturmak ve çalıştmak için birlikte çalışır ve bu üç katman web geliştirmenin temelini oluşturur.

BAŞLANGIÇ AŞAMASI GELİŞİMİLER	ARKA UÇ GELİŞİMİLER	TAM YİĞİN GELİŞİMİLER
<p>Ön yüzünü kodlayın bir internet sitesi; yani kullanıcı bölümünü görür ve etkileşime girer ile birlikte.</p> <p>Web'i getirin tasarımcı tasarımları HTML kullanarak hayatı, JavaScript ve CSS.</p> <p>Duyarlılık sağlamak tasarım.</p>	<p>Arka planda alışveriş sahneler, bina ve sürdürmek teknoloji gerekliliğini sağlamak başlangıç aşaması.</p> <p>Emin olun her şey ön uç geliştirici yapıları tamamen işlevsel.</p> <p>Oluşturun ve yönetin veritabanı.</p>	<p>Her iki alanda da uzmanlar ön uç ve arka uç gelişim.</p> <p>Strateji kılavuzu ve en iyi uygulamalar.</p> <p>Her ikisinde de iyi derecede bilgili iş mantığı ve kullanıcı deneyimi.</p>

Şekil 10. Bir web geliştiricisi ne yapar?

Web geliştirme ve web tasarım arasındaki fark Yazılım mühendisliğiinde olduğum gibi, "web geliştirme" ve "web tasarım" terimlerinin birbirinin yerine kullanıldığıını da duyabilirsiniz, ancak bunlar çok farklı şeylerdir.

Bir araba yapmak için birlikte çalışan bir web tasarımcısı ve web geliştiricisi düşünün: geliştirici motor, tekerlekler ve dişliler gibi tüm işlevsel bileşenlerle ilgilenirken tasarımcı hem görsel yönlerden hem de arabanın nasıl olacağından sorumlu olacaktır. Görünüş, gösterge panelinin düzeni, koltukların tasarımı - ve otomobilin sağladığını kullanıcı deneyimi için, yanı sorunsuz bir sürüş olsun ya da olmasın.

Web tasarımcıları, web sitesinin nasıl göründüğüünü ve hissettiğini tasarlar. Web sitesinin düzenini modeller, mantıklı, kullanıcı dostu ve kullanımı hoş olduğunu undan emin olurlar. Tüm farklı görsel unsurları göz önünde bulundururlar: hangi renk şemaları ve yazı tipleri kullanılacak? Hangi düğmeler, açılır menüler ve kaydırma çubukları dahil edilmelidir ve nerede? Web tasarımları ayrıca, hangi içeriğin dahil edileceğini ve nereye yerleştirileceğini belirleyerek web sitesinin bilgi mimarisini de dikkate alır.

Web tasarımları son derece geniş bir alandır ve genellikle Kullanıcı Deneyimi Tasarımı, Kullanıcı Arayüzü Tasarımı ve Bilgi Mimarisi gibi daha spesifik rollere bölünecektir.

Bu tasarımları alıp canlı, tamamen işlevsel bir web sitesine dönüştürmek web geliştiricisinin işidir. Bir ön uç geliştirici, web tasarımcısı tarafından sağlanan görsel tasarımları alır ve HTML, CSS ve JavaScript gibi kodlama dillerini kullanarak oluşturur. Bir arka uç geliştiricisi, bir e-ticaret sitesindeki ödeme işlevi gibi sitenin daha gelişmiş işlevlerini oluşturur.

Kısacası, bir web tasarımcısı mimar, web geliştiricisi ise inşaatçı veya mühendistir.

[https://en.wikipedia.org/wiki/Web_development]
[<https://careerfoundry.com/en/blog/web-development/>]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Bir web sitesi, statik bir web sayfası (bir sayfa önceden oluşturulmuş ve müşterilere daha sonra teslim edilmek üzere saklanmıştır), araç, karmaşık web uygulamaları, görev listesi, yürütme sunucudaki komut dosyaları, istemcilerle ilişkileri sürdürme, işbirlikçi çabalar, sorumlu departman, özel geliştirici arayüzler, tam döngü geliştirici, yazı tipi seti, istek gönder, çıkar, birbirinin yerine geç, düşeni ortaya çıkar menü, sunucuda çalıştır, smth'ye güven., istemci tarafı kodu taraf.

1.2. Aşağı İdaki eş anlamlıları metinden eşleştirin

- | | |
|-------------------|--|
| 1. kurtarmak | a) görme veya görme b) |
| 2. karmaşık | ana hat c) hazırlamak, |
| 3. yürütmek 4. | tedarik etmek d) karmaşık, |
| tasarım | birkaç unsurdan oluşan e) bileşen f) esas g) |
| 5. temel 6. | atamak, atamak h) tutmak, korumak i) proje, |
| sağlamak 7. | şema j) başarmak |
| atamak 8. bileşen | |
| 9. görsel 10. | |
| yapıllandırma | |

**1.3. Metinde geçen aşağı İdaki kelime ve deyimleri anlamları ile eşleştiriniz.
anlamlar**

1. Arka uç	a) World Wide Web'de birbirine bağlı web sayfalarının bir koleksiyonu b) bir web sitesinin ön ucunu çalıṣır durumda tutmak için gereken kodlama, stil ve eklentiler gibi tüm perde arkası dijital işlemler c) tarayıcıya girilen bir web sitesinin adresi d) web sitesinin veya uygulamanın kullanıcının gördüğü ü kismı. Web sitenizin arka ucu sahne arkasındaki her şeyse, sahnede olan budur e) son kullanıcıların bir veritabanındaki verileri oluşturmasını, korumasını, okumasını, güncellemesini ve silmesini mümkün kıyan veritabanları oluşturmak ve yönetmek için sistem yazılımı f) bir web sitesinin düzenini, görsel görünümünü ve kullanılabilirliğini tasarlamaktan sorumlu bir BT uzmanı
2. alan	
3. Web tasarımcısı	
4. web sitesi	
5. ön uç	
6. veritabanı yönetim sistemi	

1.4. Metni tekrar okuyun ve aşağıda ifadelerin doğruluğu olup olmadığına karar verin veya yanlış.

- 1) Web geliştirme, özel bir ağ içi web siteleri ve uygulamalar oluşturma sürecidir.
- 2) Web geliştirme, basit bir tek statik düz metin sayfası geliştirmekten karmaşık web uygulamalarına, elektronik işletmelere ve sosyal ağ hizmetlerine kadar doğrulanabilir.
- 3) İki tür Web geliştirici uzmanlığı vardır: ön uç geliştirici ve arka uç geliştirici.
- 4) Arka uç çok gerçek değil çünkü kullanıcı gerçekten görmüyor BT.
- 5) Web tasarımları son derece geniş bir alandır.
- 6) Arka uç geliştiricileri, ön uç geliştiricinin her şeyin yapıları tamamen işlevseldir.
- 7) Tam kapsamlı bir geliştirici, bir e-ticaret sitesindeki ödeme işlevi gibi sitenin daha gelişmiş işlevlerini oluşturur.
- 8) Bazen Web Siteleri veritabanı teknolojisine güvenmez.

1.5. Aşağıda soruları metne göre cevaplayınız

- 1) Tam yönetim geliştirici ne yapar?
- 2) Bir arka uç geliştirici nelerden sorumludur?
- 3) Bir ön uç geliştirici ne yapar?
- 4) Herhangi bir web geliştirme ürünü adlandırılabilir misiniz?
- 5) Web geliştirme ve web tasarımları arasındaki fark nedir?
- 6) Web geliştirme ekibi kaç kişiden oluşabilir?
- 7) Web geliştirme hangi görevleri içerir?

2. Dilbilgisine Odaklanın

2.1. Participle I ve Participle II tablosunu inceleyin

Tür		(Aktif)		(Pasif)	
	Şimdiki Katılımcı Basit	geliştirme - geliştirme ; ilk web sitesini geliştirirken bazı hatalar yapabilir .			geliştiriliyor - geliştirilmekte; olmak gelişmiş (genel)
	Şimdiki Katılımcı Mükemmel	geliştirmiş - geliştirmiş , (zaten, bir şeyden önce) Programı geliştiren firmamız bakımını sunabilir.			olan - (zaten) geliştirildi
	II. parçacık (Geçmiş Participle)	-----			- III gelişmiş - gelişmiş

2.2. Cümleleri okuyun ve çevirin. Participle I, II'nin işlevleri hakkında yorum yapın

- 1) Yazılım mühendisliği i, yöntemler sağlamayı amaçlayan disiplindir.
ve yazılım sistemleri geliştirme prosedürleri.
- 2) Ekapsamlı simülasyon ve prototipleme bazen
ve insanla ilgili sistem gereksinimlerini analiz etmek
etkileşim.
- 3) Üniversitenin Cockrell Mühendislik Okulundaki Araştırmacılar
Texas of Texas yeni, açık kaynaklı bir bilgisayar programlama geliştirdi
Web'i öne sürülmüş ölçüde daha fazla enerji haline getirebilecek çerçeveye
verimli, insanların gezinirken daha fazla pil gücünden tasarruf etmelerini sağlar
mobil cihazlarda.
- 4) Raporun hediyelerini işittince, Bay Smith bunu itiraz etmedi.
- 5) Bilgisayarını kapatıp terasa çıktı.
- 6) İceri girdi, şaşırıldı ama ilgilendi.
- 7) Sonunda adının seslenildiğiini duydu.
- 8) Mobil cihaz kullanıcıları zamanlarının yaklaşık üçte ikisini internette gezinerek geçiriyor
ağ .
- 9) Ve verilen her cevap, kağıtda hızlıca yazıldı.
kağıt it.

2.3. Doğru varyantı seçin

- 1) Bu makaleyi _____ aldınız mı? a) yazılı b)
yazı tipi c) tip 2) yedide _____ yetişmeyecek miyiz? a) ayrıldı b)
ayrıldı c) ayrıldı 3) Dün _____ harfi hoş karşılanmadı. a) aldı b) aldı c)
raporunu aldıktan sonra _____ (sob) hattında harc yarın da _____ Bayramda _____ in
hiç bir yere gidemeyeceğim

4) _____

mobilya _____.

- a) teslim ediyor b) teslim ediliyor c) ders ödevi için makaleler teslim etti,
6) _____ üniversiteye devam ederken gazeteci olarak _____ paraya başladı. a)
yazma, kazanma; b) yazmış olmak, kazanmak; c) Yazmış olmak, problemle nasıl başa
çıkaçığını kazanmak.
7) Yardım için bana döndü, _____
a) bilmemek b) bilmemek c) bilmemek
8) Alice bilgisayar derslerini sevmiyordu; b) sıkıcı olduklarını düşündü _____ .
a) sıkılmış c) sıkıcı

3. Tartışma

3.1. Bazı web sitelerine bakın. Aralarındaki tasarım farklılıklarını not edin. Gezinme çubukları, kullandıkları kategoriler ve animasyonlar hakkında bir rapor oluşturun. Hangi tasarım özelliklerini fark edebilirsiniz?

3.2. Aşağıdaki soruları tartışınız:

- 1) İnsanların neden kişisel web siteleri var?
- 2) Hiç birinin kişisel ana sayfasını ziyaret ettiniz mi? Nasıldı? Şirketlerin neden web siteleri var?
- 3) Bir web sitesi ile bir web sayfası arasındaki fark nedir?

4. Ek okuma

4.1. Diğ er geliştirme araçlarıyla ilgili metni okuyun ve çevirin

Diğ er web geliştirme araçları

Web geliştirme araçları (genellikle devtools veya inspect ög esı olarak adlandırılır), web geliştiricilerinin kodlarını test etmelerine ve hatalarını ayıklamalarına olanak tanır. Web sitesi oluşturuculardan ve entegre geliştirme ortamlarından (IDE'ler) farklıdır, çünkü bir web sayfasının doğrudan oluşturulmasına yardımcı olmazlar, daha ziyade bir web sitesinin veya web uygulamasının kullanıcı arayüzüne test etmek için kullanılan araçlardır.

Web geliştirme araçları, web tarayıcılarında tarayıcı eklentileri veya yerleşik özellikler olarak gelir. Google Chrome, Firefox, Internet Explorer, Safari, Microsoft Edge ve Opera gibi en popüler web tarayıcıları, web geliştiricilerine yardımcı olacak yerleşik araçlara sahiptir ve ilgili eklenti indirme merkezlerinde birçok ek eklenti bulunabilir.

Web geliştirme araçları, geliştiricilerin HTML, CSS, DOM, JavaScript ve web tarayıcısı tarafından işlenen diğ er bileşenler dahil olmak üzere çeşitli web teknolojileriyle çalışmasına olanak tanır. Web tarayıcılarından daha fazlasını yapma talebinin artması nedeniyle, popüler web tarayıcıları geliştiricilere yönelik daha fazla özellik içeriyor.

Web geliştiricileri ayrıca kodlarını yazmak için Atom, Sublime veya Visual Studio Code gibi bir metin düzenleyici kullanacak; Chrome veya Firefox gibi bir web tarayıcısı; ve son derece önemli bir araç: Git!

Git, geliştiricilerin kodlarını depolayıp yönetebilecekleri bir sürüm kontrol sistemidir. Bir web geliştiricisi olarak, kodunuzda sürekli değil işiklikler yapmanız kaçınılmazdır, bu nedenle Git gibi bu değil işiklikleri izlemenizi ve gerekirse tersine çevirmenizi sağlayan bir araç son derece değil erlidir. Git ayrıca diğ er ekiplerle çalışmayı ve aynı anda birden fazla projeyi yönetmeyi de kolaylaştırır. Git, web geliştirme dünyasında öyle bir temel haline geldi ki, artık onu kullanmamak gerçekten kötü bir uygulama olarak görülmektedir.

Bir diğ er son derece popüler araç, Git'in bir bulut arayüzü olan GitHub'dır. GitHub kılavuzumuzda ne olduğunu ve nasıl kullanılacağı hakkında daha fazla bilgi versek de, aslında bu araç Git'in tüm sürüm kontrol işlevlerini sunar, ancak aynı zamanda hata izleme, görev yönetimi ve proje wiki'leri gibi kendi özellikleriyle birlikte gelir.

GitHub yalnızca depoları barındırmakla kalmaz; ayrıca geliştiricilere kapsamlı bir araç seti sağlayarak kodlama için en iyi uygulamaları izlemeyi kolaylaştırır.

Açık kaynak projeleri için bir yer olarak kabul edilir ve ayrıca web geliştiricilerinin becerilerini sergilemeleri için bir platform sağlar.

Web sitenizin kapsamlı bir performans analizini göründülerken HTML ve CSS'nizi gerçek zamanlı olarak düzenleyebilmeniz veya JavaScript'inizde hata ayıklayabilmeniz harika olmaz mıydı?

Google'ın yerleşik Chrome Geliştirici Araçları, tam da bunu yapmanıza olanak tanır. Hem Chrome'da hem de Safari'de paketlenmiş ve mevcut olup, geliştiricilerin web uygulamalarının iç kısımlarına erişmesine izin verir. Bunun da ötesinde, bir ağ araçları paleti, yükleme akışlarınızı optimize etmenize yardımcı olabilirken, bir zaman çizelgesi, herhangi bir anda tarayıcınızın ne yaptığıını daha iyi anlamınızı sağlar.

Google her altı haftada bir güncelleme yayınlar; bu nedenle becerilerinizi güncel tutmak için web sitelerine ve Google Developers YouTube kanalına göz atın. (2500)

4.2. Ek okuma için metnin kısa bir özetini yazın

4.3. 'Web geliştirme' ve 'Web geliştirme araçları' terimlerinin kısa bir açıklamasını yapın

ÜNİ TE 10. Bir Web Sayfasının Bazı Temel Öğeleri

Öğrenme hedefleri

Bir web sayfasının temel öğelerini gözden geçirmek Her bir öğenin ne gibi bir etkiye sahip olduğunu ve buna nasıl katkıda bulunduğuunu anlamak genel kullanıcı deneyimi
Web sayfası geliştirmede kullanılan bazı bilgisayar terimlerini öğrenmek

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Başlık, sayfayı kaydirmak için, web sitesi düzeni, deneme sürümü, harekete geçirici mesaj düğmesi, gizlemek için, hamburger menüsü, yapışkan başlık, iki katmanlı gezinme, marka kimliği, dikkat çekmek için, kahraman bölümü, dikkat çekmek için, alt bilgi, kaydırıcı, carousel, dahili arama, arama sorusu, kısayol, yanlış yorumlama, daha düşük hemen çıkma oranı, daha ince, haberdar olmak, bir arama kutusu, şirket logosu, bir teknik uygulamak, web sitesi içeriği.

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.

Başlık, web sayfasının üst (üst) kısmıdır. İnsanların web sitesinde ilk saniyelerinde sayfayı kaydırmadan önce gördükleri alan olan başlık, stratejik öneme sahip bir unsurdur. Başlıktan, kullanıcıların siteyi saniyeler içinde tarayabilmeleri ve onlara yardımcı olabilecek ana sayfalara atlayabilmeleri için web sitesindeki temel gezinmeyi sağlama beklenir. Başlıklara ayrıca site menüleri denir ve web sitesi içinde birincil gezinme öğesi olarak konumlandırılır. Başlıklar bir dizi anlamlı düzen öğesi içerebilir, örneğin: marka kimliğiinin temel öğeleri, genellikle bir logo harekete geçirici mesaj düğmesi web sitesi içeriğinin temel kategorilerine bağlı lantılar sosyal ağlara bağlı lantılar temel iletişim bilgileri (telefon numarası, e-posta adresi,

vb.)

çok dilli arayüz olması durumunda dillerin de değiştircisi

arama alanı

abonelik alanı veya düğmesi

deneme sürümü gibi ürünle etkileşime bağlanılar,

AppStore'dan indirme vb.

Bir başlık i web kullanabilirliğine katkıda bulunan hayatı bir unsur yapan şey, bir web sayfasının en taranabilir bölgelerine yerleştirilmiş olmasıdır.

Kullanıcıların bir web sitesinde hangi tarama modeline bağlı kaldıkları ne olursa olsun, sayfanın üst kısmından başlar, aynı okuma ve yazma düzenini kullanan diller içinden soldan sağa taranır. Web başlıkları için popüler tasarım uygulamalarından bazıları şunlardır:

hamburger menüsü: tipik bir ekmek-et-ekmek hamburgerine benzeyen yatay çizgilerden oluşan ve içi hamburger düğmesi adı verilen farklı sayfalara veya bölgelere bağlı bağlantı setini gizleme.

yapışkan başlık: gizlenmeyen ancak kullanıcılar sayfayı aşağı kaydırırken sayfanın üst kısmına yapışan başlık. Bu şekilde, uzun kaydırılmış iç erik ağırlıklı sayfalar açısından yardımcı olabilecek herhangi bir etkileşim noktasında temel gezinme alanı kullanılabilir. İki katmanlı navigasyon: her ikisi de web kullanabilirliği için önemli olan iki farklı navigasyon rotasını ayırmak için başlıklı bir tür çift navigasyon sitesi seti.

Web sitesi başlıkları için daha yaygın olarak kullanılan bir kalıp, bir logoyu tıklanabilir yapmak ve tıklandıktan sonra ana sayfayı açmak veya yenilemektir. Nasıl çalıştığıyla ilgileniyorsanız, [https://blog.tubikstudio.com/anatomy of-web-page](https://blog.tubikstudio.com/anatomy-of-web-page) adresini ziyaret edin. Belirli bir eylem. Bu eylem, belirli bir sayfa veya ekran için bir döngümüz sunar (örneğin, satın alın, iletişim kurun, abone olun, vb.). Başka bir deyişle, pasif bir kullanıcıyı aktif bir kullanıcıya dönüştürür.

Bu düğmelerin türü, ilgi çekici doğası nedeniyle sayfadaki veya ekranındaki diğer tüm düğmelerden farklıdır: dikkat çekmeli ve kullanıcıları gerekliliği yapmaya teşvik etmelidir.

Etkili harekete geçirici mesaj düğmelerinin fark edilmesi kolaydır; tasarımcılar bunları kasıtlı olarak oluştururlar, böylece web sitesi ziyaretçileri onları saniyeler içinde görebilir ve yanıt verebilir. Bu nedenle, genellikle belirli bir harekete geçirici mesaj içeren mikro kopya içeren kalın düğmelerdir (örneğin, "Daha fazla bilgi edinin" veya "Satın alın")

Şimdi"), bu sayfa iç inana eylemine olduğ unu açıkla bunu yapmaya teşvik eder. bir kullanıcır

Bubölüm ekranın üst kısmındaki ie'dir (W web sayfasının güçlü görsel kancayı sunan
ö ğ eyi içeren ön kavurma alanı bir kahraman nöruktüsü kavdası ve kılıçın kılıçmanı
ayarlanması gibi şık ve klasik dansın tekniklerini gösteren bilgi sağlayıcı, eğitici ve etkileşime açılmış bir alan),

a

n

e kullanıcıları ile, tr

Footer :

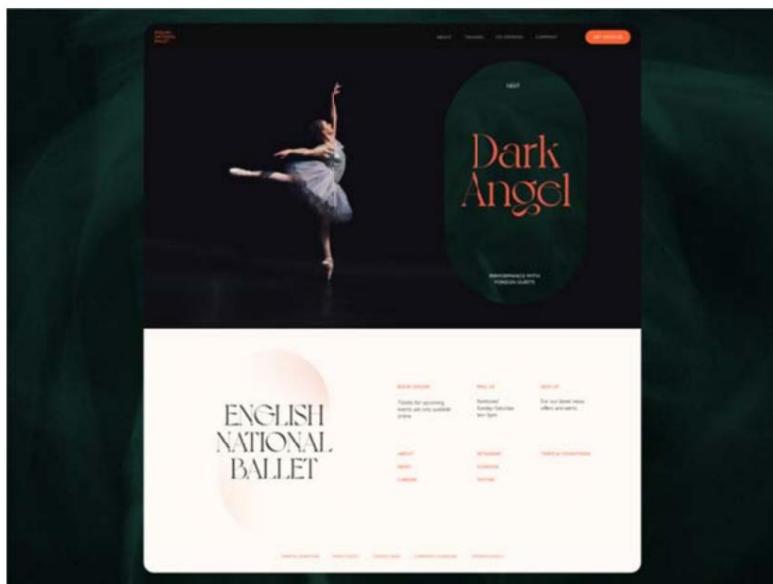
üzgünüm

,

, örneğ iFA e, Hakkı AQA sayfası t ve
, Şartlar Koşullar, , Destek Ekibi,

,

web sitesi yaratıcılarına kredi
sosyal ağda keşfetmek için benimle konuş
herhangi bir veya pr solistin paşıkların referansları ve rozet sertifika
bu



Slider , farklı ürünleri, teklifleri vb. sunmak için bir slayt gösterisi veya carousel teknigi ini uygulayan etkileşimli bir ög edir. Bir tür ürün veya hizmet galerisi sunmak için e-ticaret ve iş web sitelerinin bir parçası olarak özellikle popülerdir.

Dahili arama , bir ziyaretçinin web sitesi içindeki içeriğ e göz atmasını sağlayan ve bunu arama sorgusuna göre gösteren bir işlevdir.

Doğu ayarlandığında ilgili içeriği gösterir ve bu şekilde kullanıcının ihtiyaç duyduğu kısa yol sağlar. Böylece dahili arama, kullanıcının zaman ve çabasından tasarruf sağlar, dijital ürünün kullanılabilirliğini ve arzu edilebilirliğini artırır, kullanıcıları elde tutmaya yardımcı olur ve dönüşüm oranlarını artırır. Kullanıcı arabirimindeki dahili aramadan sorumlu olan etkileşimli öge, arama kutusu veya arama çubuğu olarak da adlandırılan bir arama alanıdır: kullanıcının arama sorgusunu yazmasını ve bu şekilde ihtiyaç duyulan içerik parçalarını bulmasını sağlar.

Web siteniz 50'den fazla sayfadan oluşuyorsa, dahili aramayı uygulamayı düşünmenin tam zamanı. İyi tasarlanmış ve kolay bulunan arama alanı, kullanıcının çok sayıda sayfa ve menü arasında gezinmeden gerekli noktaya atlamasını sağlar. Tek sayfalık bir web siteniz varsa, uygulamanız veya web siteniz özlüyse ve yoğun içerikle dolu değil ise, dahili aramaya gerek yoktur. Bir örnek daha burada <https://blog.tubikstudio.com/anatomy-of-web-page/Breadcrumbs> , kullanıcıları web sitesinde bir yolculukta desteklemek için kullanılan gezinme öge eleridir: web sitesinde nerede olduklarının farkına varırlar ve web sitesi yapısına daha kolay ulaşabilir. Bu nedenle, kıııntılar ikincil gezinme düzeyini sunar ve çok sayıda sayfası olması durumunda web sitesinin kullanılabilirliğini artırır.

İçerik kıııntılarının faydalardan bazıları

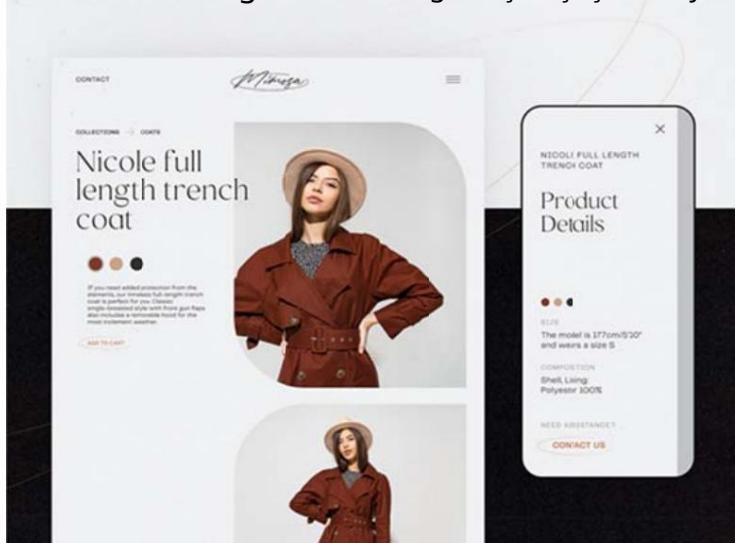
şunlardır: daha fazla bulunabilirlik daha az tıklama gerekli

düz görümlü metin öğeleriyle ekran boşluk çizgisinin etkin kullanımı
çok fazla alana ihtiyaç duymayan
yanlış yorumlama yok daha

düşük hemen çıkma oranı: içerik kıııntıları, ilk kez gelen ziyaretçiler veya karmaşık web siteleriyle uğraşma konusunda günlük deneyimi olmayan kişiler için harika bir destektir.

Dahili web sitesi aramasının yanı sıra, içerik haritaları, web sitesinin birden fazla sayfası ve birden çok katmandan oluşan karmaşık bir hiyerarşisi olduğunu yardımcı olur. Ekmek kıııntıları, büyük e-ticaret web sitelerinde ve platformlarda, medyada ve haberlerde yaygın ve kullanıcılar tarafından bekleniyor

web siteleri, bloglar, ve m dergileri çok çeşitli en iyiresimleri vb kapsıyor.



[<https://blog.tuubikstudio.com/nicole-full-length-trench-coat/> web sayfası]

1.1. Metin-babanlı atamalar

1.1.1. Takip eden kelimelerin ve nkelime mbinatio'sunu n gilizce karşılıkları

ortak

Biz:

Chapa; mg ~~mbinatio~~ şablon; komuşonbahar göz; elementi; yü "sakız
dosyası x satırı bağ lantılar dürtüsü" (~~kişimgesi~~
ile); tsamidahafazla tayfandantgördürmek bastırmak k üzerinde iiweb'i dahafazla iç erikle açan ; şanslı
kasıtlı olarak; ny kr görsel kanca; sayfa; n çektilenmak

başlayalım

atölye

yabancı ~~tepe~~inde; dikkat çektiğiydürüci al ~~bilgi~~; ekmek r;
kiler); verimlilik araması

kırıntılar ve

(yeni ulusal zincir pranga troka; kim dö nüşüm ent; ; sabah
"gereksizx görünümler" gö stergesi"; hat i ekranda; iç eri
Arama.

1.2. Ma Aşağı idaki f'yig hesaplama terimindeki mirasçı tanımları ile eşleştirin

1. Web sayfası	a) genellikle en üsttek bir web sitesinin alanı , bileme numarasını çırkati blog hizmeti sağlayıcısının .ph c iç erir.
2. Uzakta	isteyeniz hizmeti, teknolojiyi kullanarak online döküman ve hazır misiniz? Şimdi ücretsiz denemeye başlayın' Şimdi', 'sen

3. Başlık	c) genellikle HTML/XHTML ile yazılmış, bir web tarayıcısında görüntülenmesi amaçlanan tek bir belge. Çok u durumda, web sayfaları başka kodlama ve programlama da iç erir (PHP, Ruby on Rails veya ASP gibi). d) bir sayfada ne olduğunu ve nerede olduğunu açıklar, sayfa
4. Altbilgi	yapı.
5. Ekmek kırlıntıları	e) genellikle, yasal bilgiler vb., Metin Yazarlığı bilgileri vb. dahil olmak üzere dahili sayfalara bağlı lantillardan oluşan bir web sayfasının alt alanı. f) Başlığının altındaki, belirli bir sayfanın
6. Bir Harekete Geçirici Mesaj	içeren böülümlerini gösteren, genellikle 'ana sayfa > olarak gösterilen küçük bağlı lantilar kategori > alt kategori > geçerli sayfa'. Bunlar, kullanıcıların site yapısını gezinmesine ve anlamasına yardımcı olmak için mevcuttur.

1.3. Aşağıdaki soruları metne göre cevaplayınız

- 1) Başlık neleri iç erir? Bir başlığının web kullanılabilirliğine katkıda bulunan hayatı bir unsur yapan nedir?
- 2) Altbilgi, web sayfasının alt kısmı mı yoksa üst kısmı mı?
- 3) Kaydırıcıının rolü nedir?
- 4) Hangi ilke? "Kullanım kolaylığı / web sitesinin ne kadar kullanıcı dostu olduğunu Erişilebilirlik, Kullanılabilirlik, Açıklık veya İçerik?"
- 5) Dahili arama, bir ziyaretçinin ne yapmasını sağlar?
- 6) Ekmek kırlıntıları birincil veya ikincil seviyeyi mi sunuyor? navigasyon?
- 7) Bir harekete geçirici mesaj düğmesi neden diğer tüm düğmelerden farklıdır?

1.4. Metni tekrar okuyun ve aşağıdaki ifadelerin doğrusu mu yoksa yanlış mu olduğunu karar verin.

- 1) Harekete geçirici mesaj düğmesi, stratejik öneme sahip bir unsurdur.
- 2) Başlıklara site menüleri de denir ve web sitesi üzerinde birincil gezinme öğesi olarak konumlandırılır.
- 3) Web başlıklarını içeren popüler tasarım uygulamalarından bazıları şunlardır: hamburger menüsü, yapışkan başlık ve dahili arama.
- 4) İki katmanlı navigasyon, iki farklı navigasyon rotasını ayırmak için başlıklı bir tür çift navigasyon sitesidir.

- 5) Harekete geçirici mesaj (CTA) düğmesi, pasif bir kullanıcıyı aktif bir kullanıcıya dönüştürür.
- 6) Ana fikir, kahraman bölümündeki görsel kancanın yanında dikkat çekmesi ve kullanıcılarla hızlı görsel, duygusal ve bilgilendirici bağ lantı kurulmasına izin vermesidir.
- 7) Altbilgiler şunları içerebilir: harekete geçirici mesaj düğmesi, web sitesi içeriğinin temel kategorilerine bağ lantılar, sosyal ağlara bağ lantılar, temel iletişim bilgileri (telefon numarası, e-posta adresi vb.).
- 8) Dahili arama, dijital ürünün kullanılabilirliğini ve arzu edilebilirliğini artırır, kullanıcıları elde tutmaya yardımcı olur ve döşüm oranlarını artırır. Ama çok zaman alıyor.
- 9) Ekmek kırıntıları, web sitesinin birden çok sayfası ve birden çok katmandan oluşan karmaşık bir hiyerarşisi olduğunda yardımcı olur.

2. Dilbilgisine Odaklanın

2.1. Aşağıdaki yapıları Participles ile gözden geçirin

1. Hedef - ile - Participle I Yapısı (Participle ile Karmaşık Nesne)	ne zaman kullanılır konuşmacı, katılımcı tarafından ifade edilen eylemin, tamamlandı ve devam ediyor konuşma anı.	Projesi üzerinde çalıştığıını gördüm .
2. Öznel Katılımcı İnya (Karmaşık Konu)	duyusal zihinsel algı fiilleriyle birlikte kullanılır pasif yapı karakteristik yazmak için. eylemi ifade	Jane projesi üzerinde çalışırken bulundu .
3. Aday Mutlak Katılımcı İnyaat	eder, fiil tarafından belirtilen eylemle ilişkili yüklem cümleleri. Döşünün kendisi oluşur genel olarak isim vaka (daha az sıklıkla zamirler aday durumda) ve cemaat. Bu ciro yazı karakteristiği söylemek .	Makale tercüme edildikten sonra öğrenci öğretmene gösterdi. Makale tercüme edildikten (ne zaman) sonra, öğrencisi gösterdi onun öğretmeni. (durum zaman)

2.2. Katılımcı yapılarla karmaşık nesneye ve karmaşık özneye dikkat edin. Aşağı idaki cümleleri Rusçaya çevirin

- 1) Hareket ettiğini duydum ve şu anda yeni bir yazıcıya geri döndü.
- 2) Büyük boş ofisin ortasına yürüken hareketsiz kaldı.
- 3) Telefonu kaldırarak, 'Evet?' 4) Onu bir şeyler yaparken, basarken, kodlarken ve tasarlarken izlemeyi severdi. 5) Prensipte her zaman geç kalmamıştır, ilkesi dakikliğinin zaman hırsızıdır.
- 6) İçinden çıkmak istiyor gibisin.
- 7) Kağıtlarımızı kontrol ettikten sonra, kendini işe uygun hissetmiyordu.
- 8) İnsanların çok yaklaşmasını sevmiyorum.
- 9) Kural cevaplandıktan sonra, analize geçtik.
cümleler.
- 10) Hilbert sahip olduğu her şeyi kullandığı için para yoktu.
- 11) Kimsenin söyleyecek başka bir şeyi olmаяnca dışarı çıktı.
- 12) Görevin öğreniciler için çok zor olduğunu anlaşıldı.
- 13) Christian, derin bir ruhsal krize katlanıyor gibi idi.
- 14) Uçağının inişini izledik.

2.3. Parantez (bağımsız eleman) olarak Participle 1 ile kendi cümlelerinizi oluşturun.

Katılımcı 1, anlamı cümlenin tamamı veya bir kısmı hakkında bir yorum olan ifadenin baş kelimesidir.

İzin vermek - içine izin vermek

Genel olarak konuşursak - genel olarak konuşursak

göre/ dan yargılamak

Şaka bir yana - şaka yok, şaka bir yana

Kenara yaslanmak - hafifçe

söylemekten bahsetmiyorum bile - hafifçe söylemek

dikkate alarak - dikkate alarak

hakkında konuşmak / hakkında konuşmak - hakkında konuşmak, hakkında konuşmak

Örneğin, genel olarak konuşursak, gerekli tüm detayları müşterimizle tartışmak sizin görevinizdir.

3. Tartışma

3.1. Bu videoyu izleyin ve en iyi 5 web sitesi hakkında yorumlarınızı yapın https://www.youtube.com/watch?v=AmHEfTSBXiY&ab_channel=Flux

3.2. Aşağıdaki soruları tartışınız

- 1) İngilizce Web tasarımının nedir? Web tasarımında önemli olan nedir?
- 2) Web Tasarım grafik tasarım mıdır?
- 3) Bir web sitesini nasıl başlatırı?
- 4) Bir web sitesi oluşturmak için HTML ve CSS yeterli mi?
- 5) Web sitesi düzenlerinin farklı türleri nelerdir?

3.3. Aşağıdaki terimleri İngilizce olarak açıklayın

Kaydırma, deneme sürümü, çok dilli arayüz, kahraman bölüm, e-ticaret, kırıntılar, dönüşüm oranları, arama çubuğu, logo.

3.4. Sorulara vereceğiniz cevapları seçin ve bunları partnerinizle tartışın. İnternette gerekli bilgileri bulabileceğinizden emin değilseniz

Web sayfaları ve web uygulamaları test soruları

- 1) Dinamik web sitesi nedir?
 - A) Etkileşimli öğeye sahip bir web sitesi B)
Etkileşim biçimini olmayan bir web sitesi C) Çok büyük bir web sitesi 2) Çoklu web sitesi hangi web geliştirme dillerinde yazılmıştır?
 - A) C++ ve Java gibi yüksek seviyeli diller B) JavaScript gibi bazı betik dilleri ile HTML ve CSS ve PHP
 - C) Sözdeler kod 3)
 - Karma nedir (içeriği birleştiren bir uygulama)
çeşitli kaynaklardan)?
 - A) Birleştirilmiş iki veya daha fazla web sitesi B) Farklı harici kaynaklardan kodları karıştıran bir web sitesi veya uygulama kaynakları

- C) Düzgün çalışmayan bir web sitesi 4) Çerezler nedir?
- A) Bilgisayarınıza indirilen virüsler B) Bilgisayarınızda depolanan ve bir web sitesine o siteye daha önce gidip gitmediğinizizi bildiren programlar C) Bilgisayarınızda depolanan ve o siteye gidip gitmediğinizizi söyleyen metin dosyaları önce site

5) Web sitesine erişen cihazın türünü ne tanımlar?

- A) Sunucu B) A protokol C) Web tarayıcı 6) İstemci tarafı komut dosyaları ile Sunucu Stili Komut Dosyaları arasındaki fark nedir?

- A) İstemci tarafı komut dosyaları, web tarayıcısı tarafından işlenen programlardır ve sunucu tarzı komut dosyaları web sunucusu tarafından işlenir B) İstemci tarafı komut dosyaları, web sunucusu tarafından işlenen programlardır ve sunucu tarzı komut dosyaları, web sunucusu tarafından işlenir. web tarayıcısı C) İstemci tarafı komut dosyaları web sitesinin statik kısımlarını işler ve sunucu tarzı komut dosyaları web sitesinin dinamik kısımlarını işler

7) Arama motorları aldıkları web sitesi sonucunu nasıl sıralar?

- A) Web siteleri, yalnızca o web sitesinde oluşturulduğundan bu yana toplamda kaç isabetin gerçekleştiğine bağlı olarak sonuç listesinin yukarısında görünür.
- B) Web siteleri, arama sonuçları listesinin yukarısında görünür çünkü daha üst sıralarda yer almaktan web tarayıcısına ödemeye yaptı
- C) Web siteleri, arama algoritması tarafından daha önemli olduklarına karar verildiklerinden, sonuçlar listesinde daha üst sıralarda görünürler. Bu, bir sitenin popüleritesi ve diğer web siteleriyle ne kadar bağlılığı olduğunu ölçülür.

8) Bulut bilişim nedir?

- A) Bulut bilişim, hizmetleri sabit disk gibi bir aygıttan yerel olarak depolamak yerine çevrimiçi olarak depolamak ve kullanmaktadır B) Bulut bilişim, ev bilgisayarınıza internet
- C) Bulut bilişim, depolanan verilere yalnızca mobil cihazlar kullanılarak çevrimiçi olarak erişilmesidir.

4. Ek okuma

4.1. Bir Web Sayfasının diğ er ög eleriyle ilgili metni okuyun ve çevirin

Menü , kullanıcı arayüzlerindeki temel gezinme ög elerinden biridir. Arayüz ile etkileşim seçeneklerini sunan grafiksel bir kontroldür. Temel olarak, komutların listesi olabilir - bu durumda seçenekler,örneğ in "kaydet", "sil", "satın al", "gönder" vb. gibi olası eylemleri işaretleyen fiillerle birlikte sunulur. verilen arayüzde içeriğ in düzenlendiğ i kategorileri sunun ve bu, onları işaretleyen isimleri kullanmanın tam zamanı olabilir.

Menüler, arayüzde farklı konumlara (yan menüler, başlık menüleri, alt menüler vb.) ve farklı görünüm ve etkileşim yollarına (açılır menüler, açılır menüler, kayan menüler vb.) sahip olabilir.

Çeşitli web sitelerinde bulunan bazı popüler türler

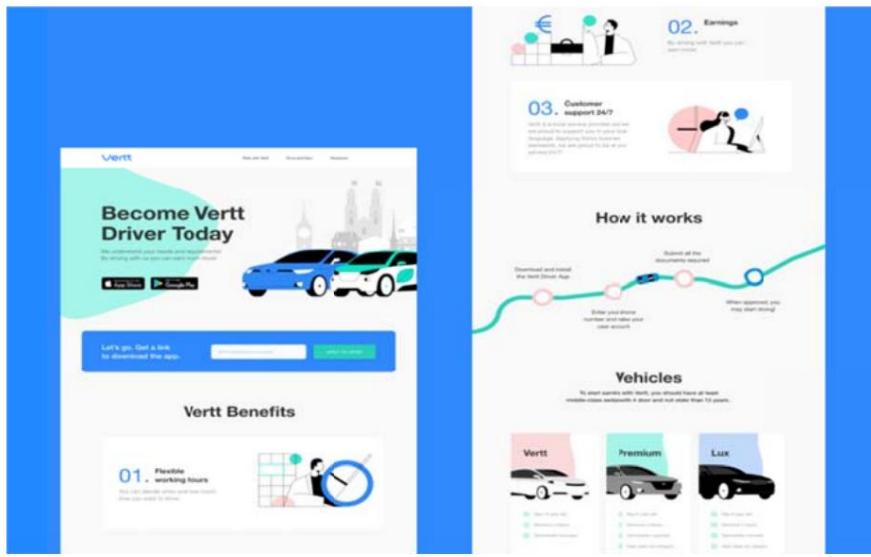
şunlardır: Klasik yatay menü: Yukarıda bahsedilen, web sitesi başlığında yatay bir çizgi olarak düzenlenen ana gezinmeyi sunan en yaygın ve iyi tanınan menü türü Kenar çubuğu menüsü: oldukça klasik bir tür, bir web sayfasının sol veya sağ tarafına yapılan dikey seçenekler listesi Açılmış menü: iç erik ağ ıraklı web siteleri için daha karmaşık bir menü türü; burada, tıkladığında veya üzerine gelindiğinde ek seçenekler listesi birincil seçenek in altında açılır. Bir başka benzer seçenek, liste açıldığıında aşağıda değil, açılır menüdür, ancak öz aynıdır.

Megamenu: Bu, birden çok seçenek in büyük listesinin iki boyutlu bir açılır menü döneminde sunulduğu karmaşık genişletilebilir menüdür; bu yaklaşım, çok sayıda seçenek in olan durumlar için etkilidir.

Hamburger menüsü: hamburger düğmesine (tipik olarak üç yatay çizgi ile işaretlenmiştir) tıkladığında menü genişler. Bu seçenek yerden tasarruf sağlar ve genellikle web sitelerinin mobil sürümlerine uygulanır. Bir örnek daha burada <https://blog.tubikstudio.com/anatomy-of-web-page/>

Form , kullanıcıların sisteme veya sunucuya bilgi göndermesini sağlayan etkileşimli bir ög edir. Özetle, birine düzenlenmiş bilgileri sağlamak için doldurmanız gereken herhangi bir gerçek kağıt formun dijital versiyonudur; ancak dijital formlar, bu süreci daha sorunsuz, net ve kullanıcı dostu hale getirmek için daha fazla seçenek ve işlevsellik sahip olabilir. Veri toplamanın geleneksel ve iyi bilinen bir yolu olduğunu undan, kullanıcılar dijital yaşamlarında formlarla oldukça sık ug raşırlar.

kayıt olma, kişisel veya finansal bilgileri eklemek, yapma, geri bildirim gönderme, e-bülten aboneliği, yenisine ödeme vb.



Formlar, kullanıcı ile dijital ürün arasındaki gerçek iletişim noktasını sunduğunu ve son derece basittir ve kullanım kolay olmalıdır. Verit Hörgası ne kadar basittir, istenilen hizmeti nasıl alabileceğimizi göstermektedir.

? Girilen verinin mantığıını düşünün, biçimsel, gezinmeyi sezgisel hale getirin ve gereklidir eylem sayısını en aza indirin.

Karo olarak adlandırılan kartlar, homog verilerinin taranabilir ve kullanıldığı sistematiğin bir şekilde düzenlenmesine ve nüfuslere uygulanmasına yardım ederler, ama her ch kart lo

gelebilir

t

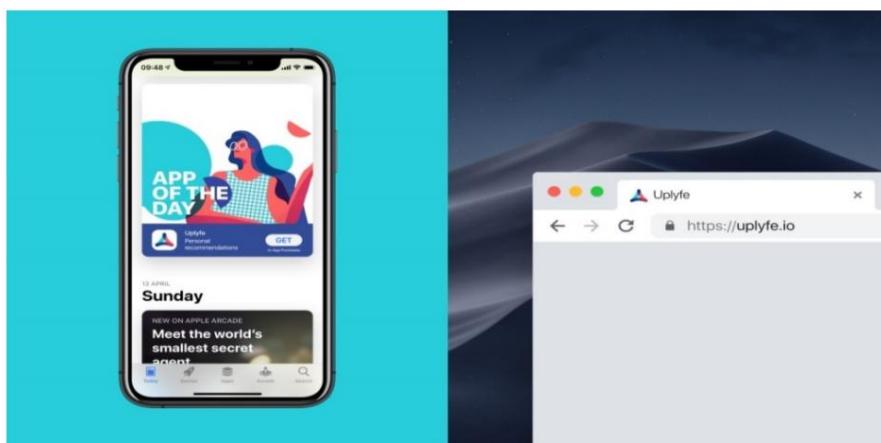


Sanat Enstitüsü blogu, yalnızca negatif boşlukla ayrılmış, ancak açıkça ayırt edilmek üzere düzenlenenmiş ultra minimalist kartlar kullanır.

Video, bir web sayfasının gerçekten temel bir parçası değil, ancak web geliştirme çözümlerinin ve teknik yeteneklerin ilerlemesiyle, bu günlerde farklı türlerde web sitelerinde giderek daha sık bulabiliyoruz. Hedef kitleyi anlayarak hazırlanmış akılda kalıcı bir video, müşterilerin dikkatini çeken bir araçtır ve onları hızlı ve parlak bir şekilde bilgilendirmenin iyi kontrol edilmiş bir yöntemidir. Video içeriği, aynı anda birkaç algı kanalını - işitsel, görsel, hareket - etkinleştirir ve genellikle bunu bir hikaye anlatımına sarılmış olarak yapar. Böyle bir faktör kombinasyonu genellikle bir video sunumunu güçlü, duygusal ve akılda kalıcı kılar.

Kullanıcıların bir ürün fikrini hızla yakalamasına, atmosferi oluşturmaya, gerekli mesajı göndermesine, hizmeti denemeye katılmasına, aracın, uygulamanın veya yazılımın nasıl çalıştığını göstermesine, kullanıcılarından gelen geri bildirimleri paylaşmasına yardımcı olan birçok başka video türüyle karşılaşabiliriz., ve saire ve saire. Ancak, yükleme süresi, kontrast sorunu, yanıt verme ve bir web sayfasına video entegrasyonu durumunda kullanıcı deneyimini bozabilecek diğer tuzaklar gibi dikkate alınması gereken önemli noktalar vardır.

URL simgesi veya yerimi simgesi olarak da bilinen Favicon, tarayıcının URL satırında ve yerimi sekmesinde ürünü veya markayı temsil eden özel bir sembol türüdür. Kullanıcıların gezinirken onunla hızlı bir görsel bağlantı kurmasını sağlar. Bu arayüz öğesi, verimli web sitesi tanıtımı ve görsel kimliğin iyi tanınması için etkili olduğunu kanıtladı. Süper küçük olması, web kullanılabilirliğine büyük katkı sağlar.



Etiketler Bu, genellikle çok sayıda homojen içeriğe sahip bloglarda ve web sitelerinde bulunan ikincil gezinme düzeyinin başka bir öğesi esidir. Etiket, kullanıcıların doğrudan öğelere atlamasını sağlayan bir anahtar kelime veya kelime öibeği ile sunulur.

onunla işaretlendi. Etiketler aslında belirli iç erik kategorilerine hızlı erişim sağlayan meta veri parçalarıdır, bu nedenle ek iç erik sınıflandırma yöntemiyle gezinmeyi desteklerler. Ayrıca etiketler genellikle kullanıcıların kendi oluşturdukları öğelerdir, bu nedenle web sitesi tarafından sabitlenen ve kullanıcılar tarafından değil istirilemeyen kategori adlarına alternatif olurlar. [<https://blog.tubikstudio.com/anatomy-of-web-page/>]

4.2. Ortakla çalışmak. Bir web Sayfasının diğer bazı öğeleri hakkında metin üzerinde birbirinize bazı sorular sorun

4.3. Aşağıdaki cümleleri İngilizce'ye çevirin

- 1) Kullanıcı arabirimini (UI) öğeleri, tasarımcıların uygulamalar veya web siteleri oluşturmak için kullanılır. 2) Kullanıcıya iletişim noktaları sağlayarak kullanıcı arayüzüne etkileşim eklerler. bunlar arasında gezinme.
- 3) Breadcrumbs (gezinme zinciri, English Breadcrumbs), site gezintisinin bir öğesi esidir; site kökünden kullanıcının şu anda bulunduğu geçerli sayfaya giden yol.
- 4) Ekmek kırıntıları genellikle üstte bir şerittir sayfanın bir kısmı, genellikle sitenin başlığıının altındadır. 5) Bildirimler, kullanıcının,örneğin bir mesaj veya bir tür sistem bildirimi gibi yeni bir şey olduğunu bilmesini sağlar. 6) Web sayfasının bir sonraki zorunlu bölümü
gezinme öğeleri, bağlantıları sağlayan köprülerdir.
bu belge sitenin diğer bölümleriyle birlikte.
- 7) Gezinme elemanları metin olarak yapılabilir çizgiler, grafik nesneleri, yani düğmeler veya etkin bileşenler.
- 8) Web sayfası bir başlangıç belgesi ise, altta böülümleri de bir ziyaretçi sayacına ev sahipliği yapıyor - küçük bir tarayıcıda belgenin her açılışını yakalayan sunucuda yüklü bir CGI komut dosyasını çağırın bir komut dosyası
sayacı gösteresinin değerini değiştirerek kullanıcılar. 9) Bu sayede web yöneticisi numarayı kolayca belirleyebilir.
herhangi bir zamanda sayfasını ziyaret eden ziyaretçiler zaman.

ÜNİ TE 11. Uygulama Geliştirme ve Uygulama Geliştirme Metodolojisi Türleri

Öğrenme hedefleri

- üygulama geliştirme hakkında temel bilgi edinmek çoğu uygulama geliştirme metodolojisinin gruplandırılabileceği üç kategoriyi dikkate almak
- üç yöntemin avantaj ve dezavantajlarını göz önünde bulundurmak
- uygulama geliştirme

Anahtar kelimeler ve ifadeler. Rusça karşılıklarını verin ve metinde kullanılan anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın.

Uygulama geliştirme, freelance geliştirici, yazılım geliştirme yaşam döngüsü (SDLC) ortaya çıkması; şelale, sıralamak, sıralamak, prototip, yönlendirmek, yerleştirmek, titiz, genç programcılar eğitmek, Hızlı Uygulama geliştirme (RAD), sprint, Çevik proje yönetimi, yüksek vasıflı, son tarih; yinelemeli, bağlı kalmak, planlı program, ihtiyaçlara uymak, eklemek.

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.



Uygulama geliştirme, yazılım uygulamaları tasarlama, oluşturma ve uygulama sürecidir. Projeler üzerinde çalışan büyük ekipleri olan büyük kuruluşlar veya tek bir serbest geliştirici tarafından yapılabilir. Uygulama geliştirme, uygulamanın nasıl yapıldığı i sürecini tanımlar ve genellikle standart bir metodoloji izler.

Projenin boyutunu, gereksinimlerin ne kadar spesifik olduğunu, müşterinin bir şeyleri ne kadar değer istirmek isteyeceğini, geliştirme ekibinin ne kadar büyük olduğunu, geliştirme ekibinin ne kadar deneyimli olduğunu ve proje için son tarihi göz önünde bulundurmalısınız.

Uygulama geliştirme, yazılım geliştirme yaşam döngüsü (SDLC) ile yakından bağlanlıdır.

SDLC'nin temel aşamaları şunlardır: Planlama, Analiz, Tasarım, İnşaat, Test, Uygulama, Destek.

Uygulama geliştirme ekiplerinin bu yedi görevi yerine getirme şekli son birkaç on yılda çok değişti ve çok sayıda uygulama geliştirme yöntemi ortaya çıktı. Her metodoloji, SDLC'nin yedi aşaması için bir çözüm sağlamalıdır.

Çoğu uygulama geliştirme metodolojisi üç kategoriden birinde gruplandırılabilir: Şelale, RAD, Çevik.

Şelale

Uygulama geliştirmede şelale yönteminin anahtar kelimeleri planlama ve sıralamadır. Tüm proje, planlama ve analiz aşamalarında haritalandırılır. Müşteri, uygulama için çok açık bir özellik ve işlevsellik listesi ile birlikte gelir. Ardından, bir proje yöneticisi tüm süreci alır ve ekip arasında haritasını çıkarır.

Bu uygulama geliştirme yöntemine şelale denir çünkü bir kez aşağı indiğinizde bir daha yukarı çıkamazsınız; her şey aşağı doğru akar. Geliştirme ekibi, belirli bir süre boyunca birlikte çalışarak tam olarak spesifikasyonlara göre sıralanılanları oluşturur. Mimari tasarlardan sonra ancak inşaat başlayabilir. Tüm uygulama oluşturuldu ve ardından düzgün çalıştığından emin olmak için hepsi test edildi.

Daha sonra müşteriye gösterilir ve uygulamaya hazır hale getirilir.

Şelale yöntemi, proje gereksinimlerinin açık olduğunu ve müşteri ile proje yöneticisinin nihai sonuç hakkında birleşik ve net bir vizyona sahip olduğunu varsayar.

Şelale yönteminin avantajı çok titiz olmasıdır.

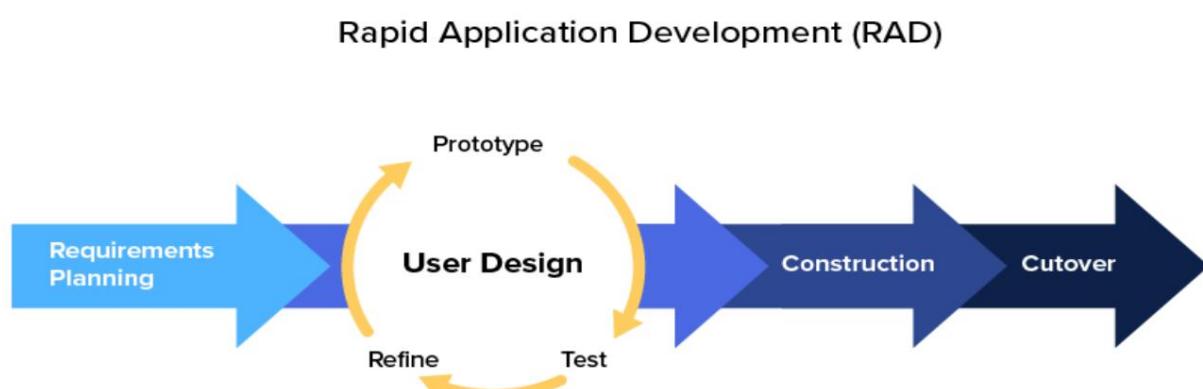
Aynı zamanda, birleştirici bir vizyona sahip olması gereken büyük projeler için kullanmak için iyi bir uygulama geliştirme yöntemidir. Şelale yöntemi aynı zamanda genç programcılar tüm projeyi onlara çevirmek zorunda kalmadan geliştirme bölümleri konusunda eğitmenin iyi bir yoludur.

Dezavantajları, değerlerin her zaman gerçekleşmesidir. Geliştirme ekibi tam olarak müşterinin istediği şeyi (ki bu her zaman olmaz), pazarı, teknolojiyi veya

organizasyon o kadar çok değil işmiş olabilir ki, etkili bir şekilde işe yaramaz ve zaman kaybıdır.

Hızlı Uygulama Geliştirme (RAD) Metodolojisi RAD, birçok yönden şelale yönteminin tam tersiydi.

RAD, çoğu unlu prototiplere dayanır, yani amaç, uygulamanın çalan bir sürümünü olabildiği ince çabuk üretmek ve ardından sürekli olarak yinelemek. Uygulama geliştirme ekibi ve müşteri, süreç boyunca birbirleriyle çok yakın çalışır. RAD ekipleri genellikle küçük ve yalnızca birkaç disiplinde yetenekli deneyimli geliştiricileri içerir. Bir projenin orijinal planından sapması gerekiyorsa, RAD bunu kolayca yerine getirebilmelidir.



Şekil 11. Hızlı Uygulama Geliştirme

RAD modelinde, her yineleme tamamlandıktan sonra ürün daha da rafine hale gelir. İlk prototipler genellikle çok kabadır, ancak ne olabileceğine dair bir resim verir. Her yineleme daha çok bitmiş ürününe benzeyir.

RAD'nin avantajları, hızlı ve oldukça esnek bir ekip ve müşteri ile çok yakın bir ilişkidir. Değişiklik bekleniyorsa, RAD bunlara şelaleden çok daha hızlı uyum sağlayabilecektir. RAD ayrıca hiçbir zaman bir prototipe fazla bağlı değildir ve onu müşterinin ihtiyaçlarına göre değiştirmeye her zaman isteklidir.

Ancak, RAD mükemmel bir uygulama geliştirme yöntemi değildir. RAD, karmaşıklığı günden güne değiştirebilecek bir proje üzerinde çalışmak için çok yetenekli (ve yüksek ücretli) programcılar gerektirir. Ayrıca son teslim tarihlerine daha az bağlılık ve teslimat tarihlerini uzatabilecek özellikler eklemeye daha fazla odaklanma var. RAD, her zaman müsait olmayabilen veya neye ihtiyaçları olduğunu bilemeyen müşterilerden çok fazla girdi gerektirir. Ek olarak,

bazı uygulamalarda, ürünün tamamını görmeden bir prototipe sahip olmak işe yaramaz.

Çevik Metodoloji

Çevik uygulama geliştirme, RAD'ye çok benzer, ancak daha büyük projelere daha uygun hale getirmek için bazı değil işiklikler de içerir. Çevik, RAD gibi yinelemeliidir, ancak özellikleri birer birer oluşturmaya odaklanır. Her özellik ekipde metodik bir şekilde oluşturulur, ancak müşteri, bir sonraki özellik geliştirilmeden önce özellikleri görmek ve bunları onaylamakla ilgilenir.

Çevik, sprintleri veya belirli bir özelliği in oluşturulması, test edilmesi ve sunulması gereken zaman kümesini kullanır. Bir özellik için tüm SDLC'yi her sprint'e dahil etmeye çalışır. Bu, ideal olarak, planlı bir programa bağlı kalmaya yardımcı olur, ancak aynı zamanda sık incelemelere de izin verir.

Çevik, prototiplere odaklanmaz, ancak tamamlanmış çalışmaları yalnızca sprint bittikten sonra sunar. Bu nedenle, müşteri şelaleden daha sık bilgilendirilirken, müşteri RAD'nin aksine yalnızca bitmiş işi görür.

Çevik proje yönetimi metodolojisi de daha çok takım veya kadro tabanlıdır. RAD ile doğrudan bir programcı ile çalışıyorsunuz. Agile ile uygulama geliştirme ekibinde ayrıca testçiler, UX tasarımcıları, teknik yazarlar ve daha pek çok kişi yer alacak.

[[https://kissflow.com/low-code/rad/types-of-application development-methodologies](https://kissflow.com/low-code/rad/types-of-application-development-methodologies)den kısaltılmıştır]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

Yazılım uygulamalarının gerçek ekleştirilmesi; serbest geliştirici; Gereksinimler; standart metodoloji; görevi gerçek ekleştirmek için; başvurmak kategorilerden biri planlanacak; detaylı liste; fonksiyonlar uygulamalar; proje Müdürü; spesifikasyonlara göre; belirli bir süre; zaman kaybı; hızlı

Uygulama geliştirme; çalışan sürüm; müşteri; sapma; bir prototipe bağlanması; ihtiyaçlara göre; Grup odaklı; planlanmış program; esnek yöntem

uygulama geliştirme.

1.2. Aşağı İdaki bilgisayar terimlerini tanımlarıyla eşleştiriniz.

1.RAD	a) yazılıma doğ rusal, sıralı bir yaklaşım yazılım mühendisliği ve ürün geliştirmede popüler olan geliştirme yaşam döngüsü (SDLC)
2.yazılım testi b) bir tekrarlanabilir sabit zaman kutusu	Mümkün olan en yüksek değer sahip "Bitti" ürünü oluşturulur c) Çevik yazılım geliştirmenin bir biçimi
3. sürat koşusu	hızlı prototip sürümlerine ve yinelemelere öncelik veren metodoloji. d) gerçek yazılımın olup olmadığıını kontrol
4. uygula	etmek için bir yöntem ürün beklenen gereksinimleri karşılar ve yazılım ürününün kusursuz olmasını sağlar. Igili bir veya daha fazla Özelliği değerlendirmek için manuel veya otomatik araçlar kullanılarak yazılım/sistem bileşenlerinin yürütülmesini içerir f) bir kod öğesini veya bir öğeyi tanımlamak ve kullanmak için.
5. prototip	programa yazılan programlama kaynağı g) orijinal bir model, form veya bir örnek
6. şelale yöntem	diğer süreçler için bir temel görevi görür. Yazılım teknolojisinde bu terim, mevcut bir ürünün yeni bir modelinin veya yeni bir versiyonunun türetilebileceği çalışmaların bir örnekidir.

1.3. Aşağı İdaki eş anlamlıları metinden eşleştirin 1.

- karmaşıklık a) ayrıntı, çok dikkatli
- 2. tam b) uygun, uygun
- 3. uygun c) akıllı, aktif, hızlı
- 4. dahil etmek d) denk gelmek, denk gelmek
- 5. uygun şekilde e) tam, eksiksiz
- 6. yönetim f) birleştirmek, birleştirmek, karıştırmak, birleştirmek
- 7. çevik g) yönetim, rehberlik
- 8. varsaymak h) karmaşıklık, karışıklık, dahil karakter
- 9. titiz i) varsaymak

1.4. Aşağıdaki soruları metne göre cevaplayınız

- 1) Uygulama geliştirmenin nasıl yapıldığıına giren faktörler nelerdir? yapıldı?
- 2) Yazılım geliştirme yaşam döngüsü kaç aşamadan oluşur?
- 3) Neden uygulama geliştirme yöntemlerinden biri olarak adlandırılıyor? şelale?
- 4) RAD daha çok prototiplere mi dayanıyor, değil mi? mükemmel mi uygulama geliştirme yöntemi?
- 5) Çevik uygulama geliştirme ne işe yarar? odaklanıyor mu prototipler?
- 6) Hızlı Uygulama Geliştirmenin amacı nedir?

1.5. Uygulama geliştirmede farklı yöntemlerin avantajları ve dezavantajları hakkındaki bilgileri kullanarak aşağıdaki tabloyu tamamlayın.

Gerekirse İnternet'ten bazı ek bilgiler kullanabilirsiniz.

	Avantajları	Dezavantajları
Şelale	1. Titiz, büyük projeler için uygundur. 2. ...	1. Değişiklikler her zaman olur. 2....
RAD metodolojisi		
Çevik metodoloji		

2. Dilbilgisine Odaklanın

2.1. Aşağıdaki tabloyu inceleyin ve farklı işlevlerde kullanılan Gerunds ile kendi cümlelerinizi oluşturun.

Gerund'un İşlevleri	Önek	Tercüme
Ders	<u>Ürünün tamamını görmeden</u> bir prototipe sahip olmak işe yaramaz.	Bir prototipe sahip olmak işe yaramaz eger bütünü görmüyorsan ürün.
Birleştirmek yüklem	Müzakereler henüz bitmiş <u>olmaktan çok uzak</u> . Müşterilerimizin <u>ihtiyaçlarını</u> tartışmaya başladık.	Müzakereler henüz değil bitti. tartışmaya başladık bizim ihtiyaçları müşteriler.

Nesne	Bilgisayarımı <u>kapattığ</u> <u>imi</u> hatırlıyorum.	kapattığ imi hatırlıyorum bir bilgisayar.
Bağ lanmak	Çalışma saatlerinin <u>azaltılması</u> önerisi şimdi tartışılıyor.	Azaltma teklifi şimdi çalışma haftası tartışılıyor.
Zarf değ iştirici	Cihazı test etmeden kullanamayız.	kullanamayız bu cihaz onsuz kontrol eder.

2.2. Cümleleri okuyun ve çevirin. Gerund'ların işlevleri hakkında yorum yapın

- 1) Müdürle konuşmanın faydası yok.
- 2) Şehrin bu bölümünde park yeri bulmak gerçek ekten zor.
- 3) Her zaman iyi bir programcı olmayı hayal etmiştir.
- 4) Hata yapmadan ög renemezsınız.
- 5) Böyle bir sorunu çözmek kolay bir iş değil.
- 6) Polisten bakmasını istemeden önce hastaneleri aramanızı öneririm.
onun için.
- 7) Köpeğ ini kaybetmeyi atlatması uzun zaman aldı.
- 8) Tom boş zamanını hayır kurumuna bağ İslamaktan gurur duyuyor.
- 9) Çevik, RAD gibi yinelemelidir, ancak özellikleri birer birer oluşturmaya odaklanır.
zaman.

2.3. Doğru varyantı seçin

- 1) Devam etmelisiniz _____ nasıl olduğunu anlayana kadar bilgisayar
_____ tüm programlar. a)
uygulama, kullanmak; b) uygulama, kullanma; c) pratik yapmak,
kullanmak 2) İzin verir misiniz a) alıyorsunuz b) önemlidir c) hatırlıyorum
asla unutmayacağım. Təqribən başarısızlığı. a) almak b) alınmış c)
almak 4) Kişiler faturalarını _____ geciktirirse, sadece giderek daha fazla faize
maruz kalırlar. a) ödeme b) ödeme c) ödeme 5) Olağanüstü hikayeden
sonra ikimiz de bir süre sessizce oturduk. a) dinlemek b) dinlemek

_____ buna

c) dinlendikten

- 6) Bir dahaki sefere kasabaya geldiğimde seni _____ dört gözle bekliyorum. Bir araya gelmeyi planlayabilmem _____ almak istiyorum. _____ görmek bilmek _____ olacak. Ondan alması 8) Bitirdi

 - b) görmek, bilmek; c) Görmek, kötü bir iz
 - _____ bilmek kimseyi şaşırtmadı. c) almış olmak
 - _____ b) İnternetten bu dosyanın alınması. c) indirmiş
 - a) indiriliyor b) indiriliyor olmak

3. Tartışma

3.1. Uygulama geliştirme ile ilgili aşağıdaki soruları tartışın

- 1) Yazılım testi neden önemlidir?
 - 2) Başvurunuz nasıl para kazandıracak?
 - 3) Web uygulamalarının bakımı nispeten kolaydır. neden olduğunu açıklayabilir misin öyle mi?
 - 4) Kullanıcılar farklı web tarayıcılarıyla etkileşime girer ve sonuç olarak, bir ürün yol haritası oluşturmak için kullanılan kullanım kalıpları ve performans ölçütlerini toplamak daha zordur. Web Uygulamaları için avantaj mı yoksa dezavantaj mı?
 - 5) Bir geliştirme ekibinde nelere dikkat etmelisiniz?
 - 6) Bir Uygulama geliştirmenin maliyeti nedir?

3.2. Aşağıdaki konuya ilgili bir dakikalık bir konuşma hazırlayın ve bunu sınıf

'En iyi fikirler, uygulanamazlarsa pek işe yaramazlar. İyi uygulama tasarımları tek başına yeterli değil; verimli, yüksek kaliteli geliştirme de gereklidir'

3.3. Uygulama geliştirme yöntemlerinden biri hakkında kısa bir rapor hazırlayın

4. Ek Okuma

4.1. World Wide Web görüntüleme dilleriyle ilgili aşağı idaki metni okuyun ve çevirin

HTML

World Wide Web, Internet üzerinden alınan metin, grafik ve sesi bir bilgisayar monitöründe görüntülemek için bir sistemdir. Her bir geri alma birimi bir Web sayfası olarak bilinir ve bu tür sayfalar sıkılıkla, alınmasıyla ilgili "bağ lantıları" iç erir.

O

izin vermek

sayfalar

HTML (köprü metni biçimlendirme dili), Web sayfalarını kodlamak için kullanılan biçimlendirme dilidir. 1980'lerde İsviçre'deki CERN nükleer fizik laboratuvarında Tim Berners-Lee tarafından tasarlandı ve bir SGML DTD tarafından tanımlandı. HTML işaretleme etiketleri, başlıklar, paragraflar ve tablolar gibi belge öğelerini belirtir. Web tarayıcısı olarak bilinen bir bilgisayar programı tarafından görüntülenmek üzere bir belgeyi işaretlerler. Tarayıcı, başlıkları, paragrafları ve tabloları ekran boyutuna ve mevcut yazı tiplerine uyarlanmış bir düzende görüntüleyerek etiketleri yorumlar.

HTML belgeleri ayrıca, diğer Web sayfalarına bağ lantıları belirten etiketler olan çapaları da iç erir. Bağ lantı, Britannica Ansiklopedisi biçimindedir; burada alınılan dize, bağ lantının işaret ettiği URL'dir (tek biçimli kaynak bulucu) (Web "adresi"). " ve onu izleyen metin, bir Web tarayıcısında görünen, başka bir sayfaya bağ lantı olduğunu göstermek için altı çizili metindir. Tek bir sayfa olarak görüntülenenler, bazıları metin ve diğerleri grafik içeren birden çok URL'den de oluşturulabilir.

XML

HTML, yeni metin öğelerinin tanımlanmasına izin vermez; yani genişletilemez. XML (genişletilebilir biçimlendirme dili), Web'de yayınlanan belgelere yönelik basitleştirilmiş bir SGML biçimidir. SGML gibi XML de belge türlerini ve bunlarda kullanılan etiketlerin anlamlarını tanımlamak için DTD'leri kullanır. XML, belge varlıklarının <BEGIN>...</BEGIN> gibi hem başlangıç hem de bitiş etiketiyle işaretlenmesi gibi ayırtırmayı kolaylaştırın kuralları benimser. XML, iki yönlü bağ lantıları ve bir belge alt bölümüne göre bağ lantıları gibi HTML'den daha fazla türde köprü metni bağ lantısı sağlar.

Web komut

dosyası oluşturma HTML veya XML ile işaretlenmiş Web sayfaları büyük ölçüde statik belgelerdir. Web komut dosyası, bir okuyucunun kullandığı gibi bir sayfaya bilgi ekleyebilir veya okuyucunun, örneğin çevrimiçi bir işletmenin sipariş departmanına iletilebilecek bilgileri girmesine izin verebilir. CGI (ortak ağ geçidi arabirim) bir mekanizma sağlar; okuyucunun Web tarayıcısı ile sayfayı sağlayan Web sunucusu arasında istekleri ve yanıtları iletilir. Sunucudaki CGI bileşeni, tarayıcı sisteminden bilgi alan veya görüntülenmesini sağlayan komut dosyaları adı verilen küçük programlar içerir. Basit bir komut dosyası okuyucunun adını sorabilir, okuyucunun kullandığı sistemin Internet adresini belirleyebilir ve bir selamlama yazdırabilir.

Komut dosyaları herhangi bir programlama dilinde yazılabılır, ancak genellikle basit metin işleme rutinleri oldukları için PERL gibi komut dosyası dilleri özellikle uygundur.

Başka bir yaklaşım, tarayıcı tarafından yürütülecek Web komut dosyaları için tasarlanmış bir dil kullanmaktadır. JavaScript, Netscape Communications Corp. tarafından tasarlanan ve hem Netscape'in hem de Microsoft'un tarayıcılarıyla kullanılabilen böyle bir dildir. JavaScript, Java'dan oldukça farklı, basit bir dildir. Bir JavaScript programı, bir Web sayfasına <script dili="JavaScript"> HTML etiketiyle katıştırılabilir. Bu etiketi takip eden JavaScript talimatları, sayfa seçildiğiinde tarayıcı tarafından yürütülecektir. Dinamik (etkileşimli) sayfaların görüntülenmesini hızlandırmak için JavaScript, sunucu ile istemcinin tarayıcısı arasında bilgi alışverişi için genellikle XML veya başka bir dil ile birleştirilir. Özellikle, XMLHttpRequest komutu, sunucunun tüm Web sayfasını yeniden göndermesine gerek kalmadan sunucudan gelen zaman uyumsuz veri isteklerini etkinleştirir. Bu yaklaşıma veya programlama "felsefesine" Ajax (eşzamansız JavaScript ve XML) adı verilir.

VB Script, Visual Basic'in bir alt kümesidir. Başlangıçta Microsoft'un Office program paketi için geliştirilmiş, daha sonra Web komut dosyası için de kullanılmıştır. Yetenekleri JavaScript'inkine benzer ve aynı şekilde HTML'ye gömülebilir.

Web programlama için bu tür betik dillerinin kullanılmasının arkasında, programların önceden yazılmış bağımsız bileşenlerin başka herhangi bir dil işleme olmaksızın birleştirilmesiyle oluşturulduğu bileşen programlama fikri yatmaktadır. JavaScript ve VB Script programları, bilgileri nasıl görüntülediklerini kontrol etmek için Web tarayıcılarına eklenebilecek bileşenler olarak tasarlanmıştır.

4.2. Ek okuma iç in metnin iç eriç i hakkında beş soru sorun

4.3. Aşağı idaki cümleleri ç evirin

- 1) Uygulama iç i satın alma, abonelik veya premium sürüm iç in ö deme yapma, reklam yerleştirme, veri satma - tüm bu yollar para kazanma özelliç ini kullanınız bile uygulamayı dağ itmak ücretsiz değil ildir.
- 2) İç mimari, cep telefonunun işlevsellig ine bağlı lıdır. uygulama ve seçilen işleme ve depolama yöntemi veri.
- 3) Genellikle iki liste yaparız:
bir uygulamaya ve anahtar görsele sahip olmalıdır elementler. Tüm geleceğ in temeli olurlar mimari eser.
- 4) Her sprintin sonunda, sonuçları aşağı idaki kişilerle tartışın.
İlgili taraflar.
- 5) Bir mobil uygulamanın geliştirilmesi, bir mobil uygulamanın geliştirilmesi ile bitmez. mağazalarda yayınalanması. Orta derecede popüler iç in bile Uygulamaların tam bir güncelleme geçmiş vardır. 6) Akıllı telefonlar, tabletler ve diğer mobil cihazlar iç in mobil uygulamalar geliştirme kavramı, yazı yazmayı içerir. sağlayacak programlar oluşturmak iç in program kodu belirli mobil platformlarda çalışın. 7) Bugün 2 ana platform var
mobil işletim sistemleri - Android ve iOS ve daha az popüler Windows Phone ve Symbian).

ÜNİ TE 12. Oyun Motoru

Öğrenme hedefleri:

Oyun motorları ve türleri hakkında temel bilgiler edinmek Oyun motorlarının neler sağladığını düşünmek

Anahtar kelimeler ve ifadeler. Rusça eşdeğ erlerini verin ve anahtar kelimelerin ve ifadelerin anlamlarını hatırlayın

Oyun motoru, yazılım çerçevesi, oyun oluşturmak için, uygulayıcı, araç takımı, işleme motoru, "ara katman yazılımı", platform soyutlama, bileşen tabanlı bir mimari, oluşturulacak, "grafik motoru", modası geçmiş, bir oyun uygulaması, çoklu platform, yapay zeka, basitleştirme, çalışma, özel bir motor, sahne grafiği, eğitim simülasyonu

Aşağıdaki metni okuyunuz ve ardından verilen alıştırmaları yapınız.



Oyun motoru, öncelikle video oyunlarının geliştirilmesi için tasarlanmış bir yazılım çerçevesidir ve genellikle ilgili kitaplıklarını ve destek programlarını içerir. "Motor" ~~çalıştırılış işleme yazılımları~~ "yazılım motoru" terimine benzer.

Oyun motoru, genellikle oyun geliştirmek için bir dizi araç ve özellik sunan bu çerçeveyi kullanan geliştirme yazılımına da atıfta bulunabilir.

Geliştiriciler, video oyun konsolları ve diğer bilgisayar türleri için oyunlar oluşturmak için oyun motorlarını kullanabilir. Tipik olarak bir oyun motoru tarafından sağlanan temel işlevsellik, 2D veya 3D grafikler için bir işleme motoru ("renderer"), bir fizik motoru veya çarpışma algılama (ve çarpışma yanıtı), ses, komut dosyası oluşturma, animasyon, yapay zeka,

sinematikler için ağ oluşturma, akış, bellek yönetimi, iş parçası olurma, yerelleştirme desteği, sahne grafiği ve video desteği.

Oyun motoru uygulayıcıları, genellikle aynı oyun motorunu farklı oyunlar üretmek veya oyunları birden fazla platforma taşımaya yardımcı olmak için yeniden kullanarak/uyarlayarak oyun geliştirme sürecinden tasarruf sağlar.

Çoğu durumda oyun motorları, yeniden kullanılabilir yazılım bileşenlerine ek olarak bir dizi görsel geliştirme aracı sağlar. Bu araçlar, oyunların veri odaklı bir şekilde basitleştirilmiş, hızlı bir şekilde geliştirilmesini sağlamak için genellikle entegre bir geliştirme ortamında sağlanır. Oyun motoru geliştiricileri, genellikle bir oyun geliştiricisinin bir oyun oluşturmak için ihtiyaç duyabileceği birçok öğeyi içeren sağlam yazılım paketleri geliştirerek uygulayıcının ihtiyaçlarını önlemeye çalışır. Çoğu oyun motoru paketi, grafik, ses, fizik ve yapay zeka (AI) işlevleri gibi geliştirmeyi kolaylaştıran olanaklar sağlar. Bu oyun motorlarına bazen "ara katman yazılımı" denir, çünkü terimin iş anlamında olduğu gibi, bir oyun uygulaması geliştirmek ve maliyetleri düşürürken ihtiyaç duyulan tüm temel işlevleri kutudan çıkar çıkmaz sağlam esnek ve yeniden kullanılabilir bir yazılım platformu sağlar., karmaşıklıklar ve pazara sunma süresi - son derece rekabetçi video oyunu endüstrisindeki tüm kritik faktörler.



12. Quake ekranı

Diğer ara katman yazılımı türleri gibi, oyun motorları da genellikle platform soyutlaması sağlar ve aynı oyunun çeşitli platformlarda (oyun konsolları ve kişisel bilgisayarlar dahil) çalışmasına izin verir ve varsa oyun kaynak kodunda çok az değişim yapılmıştır. Çoğu zaman, programcılar oyun motorlarını, motordaki belirli sistemlerin daha fazla sistemle değil istirilmesine veya genişletilmesine izin veren bileşen tabanlı bir mimariyle tasarlar.

özel (ve genellikle daha pahalı) oyun ara katman yazılımı bileşenleri. Bazı oyun motorları, daha yaygın olan entegre ürünü genişletme veya özelleştirme yaklaşımı yerine, özel bir motor oluşturmak için seçilerek birleştirilebilen, gevşek şekilde bağlanmış bir dizi oyun ara katman yazılımı bileşeni içerir. a esnek Bununla birlikte, genişletilebilirlik, uygulandıkları çok eşitli kullanıcılar nedeniyle oyun motorları için yüksek bir öncelik olmaya devam etmektedir. "Oyun motoru" adının özgüllüğünü men, son kullanıcılar genellikle oyun motorlarını, pazarlama demoları, mimari görselleştirmelerde tamamen teknolojiyi genetik hizmetlerini sağlayarak, grafik uygulamalar için yeniden amaçlar.

Bazı oyun motorları, oyunların ihtiyaç duyduğu geniş işlevsellik yelpazesi yerine yalnızca gerçek zamanlı 3B oluşturma yetenekleri sağlar. Bu motorlar, bu işlevsellikin geri kalanını uygulamak veya diğer oyun ara yazılım bileşenlerinden birleştirmek için oyun geliştiricisine güvenir. Bu tür motorlara genellikle daha kapsamlı "oyun motoru" terimi yerine "grafik motoru", "oluşturma motoru" veya "3D motor" denir. Bu terminoloji, birçok tam özellikli 3D oyun motoruna basitçe "3D motorlar" olarak atıfta bulunulduğundan, tutarsız bir şekilde kullanılmaktadır. Grafik motorlarına örnekler: Crystal Space, Genesis3D, Irrlicht, OGRE, RealmForge, Truevision3D ve Vision Engine.

Modern oyun veya grafik motorları genellikle bir sahne grafiği sağlar - genellikle oyun tasarımını basitleştiren ve geniş sanal dünyaların daha verimli bir şekilde oluşturulması için kullanılabilen 3B oyun dünyasının nesne yönelimli bir temsili.

Teknoloji yaşlandıkça, bir motorun bileşenleri belirli bir projenin gereksinimleri için eski veya yetersiz hale gelebilir. Tamamen yeni bir motor programlamadan karmaşıklığı istenmeyen gecikmelere neden olabileceğiinden (veya bir projenin baştan yeniden başlatılmasını gerektireceğiinden), bir motor geliştirme ekibi mevcut motorunu daha yeni işlevler veya bileşenlerle güncellemeyi seçebilir.

[https://en.wikipedia.org/wiki/Game_engine]

1. Metin Tabanlı Ödevler

1.1. Aşağıdaki kelimelerin ve kelime kombinasyonlarının İngilizce karşılıklarını verin:

İlgili kütüphaneler; oyun konsolu; mekanizma
görselleştirme; grafik motoru; çoklu kullanım; destek
yerelleştirme; uygulama uzmanı, geliştirici; yol
veri yönetimi; önceden; güvenilir yazılım; çapraz platform
ÜZERİ NDE; Market zamanı; oyun kaynak kodu; standart dışı
motor; bir dizi bileşen iç erir; her seferinde farklı, tutarsız; eğitim simülasyonu;
istenmeyen
gecikmeler; esnek entegre ürün; Değistirilmek; daha verimli iletim.

1.2. Aşağıdaki ön ekler ve son ekler ne anlama geliyor? Bu kelimeleri metinden
çevirin ve konuşmanın bir bölümünü belirleyin

İşlevsellik, yetersiz, etkileşimli, istenmeyen, yeniden kullanılabilir, uygulayıcı,
karmaşıklık, seçici, yeniden başlat, durdurulan, şu anda, bilgilendirici, süresiz.

1.3. Aşağıdaki ekonomik terimleri tanımlarıyla eşleştiriniz.

1. Bir grafik motoru	a) karmaşık bir yazılım sisteminin temel bir bileşeni. b) önerilen bir mimari tasarımın
2. Rekabetçi	ozelliklerini gösteren üç boyutlu görüntüler veya animasyonlar yaratma sanatı. c) bilgisayar oyunlarında kullanılmak üzere yazılımda tipik
3. Bir sahne grafiği	olarak uygulanan bir fizik modeli d) diğerlerinden daha başarılı olmayı şiddetle arzulayan;
4. Görselleştirme	karşılaştırılabilir nitelikteki diğerleri kadar iyi veya onlardan daha iyi e) bir mesajı iletmek için görüntü, diyagram veya animasyon oluşturmaya
5. Bir oyun motoru	yönelik herhangi bir teknik f) vektör tabanlı grafik düzenleme uygulamaları tarafından yaygın olarak
6. Mimari render,	kullanılan genel bir veri yapısı ve

mimari çizim veya görselleştirme	modern bilgisayar oyunları
--	----------------------------

1.4. Metni tekrar okuyun ve aşağıda idaki ifadelerin doğrusu mu yoksa doğrusu mu olduğunu karar verin.

yanlış

- 1) "Motor" terminolojisi terime tamamen benzemiyor yazılım endüstrisinde kullanılan "yazılım motoru".
- 2) Geliştiriciler, video oyunu için oyollar oluşturmak için oyun motorlarını kullanabilir konsollar ve diğer bilgisayar türleri.
- 3) Tipik olarak bir oyun motoru tarafından sağlanan temel işlevsellik, 2B veya 3B grafikler için "3B motor" içerebilir.
- 4) Çok u durumda, oyun motorları, yeniden kullanılabilir yazılım bileşenlerine ek olarak bir dizi görsel geliştirme aracı sağlar.
- 5) Programmanın karmaşıklığı, günümüzün zorlu video oyunu endüstrisindeki tek gereksinimdir.
- 6) Modern oyun veya grafik motorları genellikle bir sahne grafiği sağlar - 3B oyun dünyasının nesne yönelimli bir temsili.
- 7) Son kullanıcılar, oyun motorlarını genellikle gerçek zamanlı grafik gereksinimleri olan diğer etkileşimli uygulamalar için yeniden amaçlar.
- 8) Oyun motoru uygulayıcıları, farklı oyular üretmek istedikleri için oyun geliştirme sürecinden asla tasarruf etmezler.

1.5. Aşağıda idaki soruları metne göre cevaplayınız

- 1) Oyun motoru nedir?
- 2) Bir oyun motoru neleri içerir?
- 3) Ne tür motorlara "grafik" motorları denir? Onlardan herhangi bir örnek verebilir misiniz?
- 4) Oyun motorları ne sağlar?
- 5) Bir karaktere nasıl gerçekçilik ekleneceğini, grafik efektlerinin nasıl ekleneceğini veya bir hareketli grafiğin nasıl canlandırılacağıını bilmek gerekli midir? Yoksa bunların hepsi oyun motoru tarafından mı hallediliyor?
- 6) Yalnızca küçük bir 2D proje hedefliyorsanız, muhtemelen ihtiyacınız olmayan birçok özellik ile gelecek olan tıknaz bir motora ihtiyacınız yoktur. Bu açıklamaya katılıyor musunuz?

7) Motorların teknoloji olarak gelişimi hakkında neler söyleyebilirsiniz?
yaşlar?

2. Dilbilgisine Odaklanın

2.1. Aşağıdaki cümlelerde verilen -ing formlarının Participles veya Gerunds olup olmadığıını belirtin. Participles durumunda, niteledikleri isim veya zamiri tanımlar. Cümleleri Rusçaya Çevir

- 1) Birden fazla programı ve kullanıcıyı desteklemek ana bilgisayarın işlevidir.
işletim sistemleri.
- 2) HTML programlamayı öğrenmeniz gerekmeyen web sayfaları tasarlamak.
- 3) Yazmanın nadiren yapıldığı özel amaçlı bellek vardır.
gerekli.
- 4) Programlama, özülmesi gereken problemin analiz edilmesini içerir.
- 5) Aktarılan veriler büyük önem taşımaktadır.
- 6) Seminerimizin amacı programlamanın temel aşamalarını incelemektir.
- 7) Howard Aiken, standart makine bileşenlerini kullanarak tam otomatik bir hesap makinesi tamamladı.
- 8) Aritmetik problemini çözterken bilgisayar arızalandı.
- 9) Tasarımcı, uygun CAD yazılımını kullanarak nesne üzerinde eşitlik analizler yapabilir.

- 10) Web, web sayfalarını kullanıcıların kullanımına sunan bir İnternet hizmetidir.
milyonlarca kullanıcı.

2.2. Aşağıdaki tabloyu inceleyin Koşullar

	eğer	şart	sonuç
1 tip		Geniş zaman	irade + temel fiil
	Eğer Eğer	Tara yarın serbest, sınavlarını geçemiyorlar, Durumun gerçekleşmesi için gerçek bir olasılık var .	onu davet edecek. öğretmenleri üzülecek.
2 tip		Basit geçmiş ,	olur + temel fiil
	Eğer Eğer	önümüzdeki Temmuz kar yağdı, piyangoyu kazandım, Durumun gerçekleşmesi için gerçek dişi bir olasılık var .	Şaşırır mısın? Araba satın alayım.

3 tip		Geçmiş zaman	+ geçmiş katılımcıya sahip olurdu
	Eğer Eğer	Tara dün özgürdü, dün yağmur yağdı, Hem koşul hem de sonuç artık imkansız.	Onu davet ederdim. ne yapardın?

2.3. Koşullar oluşturmak için parantezleri açın

- 1) Sınav _____ (değil / çalışma) daha sıkı, biz _____ (geçemedik)
yaparsak. (1)
- 2) Öğrenci (geçer) ise. (3) _____ (Olmaz) sınava geç kalırlar,

- 3) Eğer (2) _____ (yanında) dizüstü bilgisayarı var, o _____ (Bana e-posta).
- 4) Eğer o _____ (gitmeyin) toplantıya, ben de _____ (gitmeyin). (1)
- 5) Öğretmen bu hafta sonu bize çok fazla ödev _____ (verirse), ben _____ (olmam)
mutlu olurum. (3)
- 6) Yeni bir araba _____ (istiyorum), bir tane _____ (satın alırım). (2)
Biz _____ (yoksa) çok fazla argümanınız varsa _____
(olma) çok inatçı! (2)
- 7) Ben (tanışmam) Amanda, eğer ben 8) Eğer _____ (gitmeyin) partiye. (3)
_____ 9) Eğer biz 10) Eğer (var ~~medeniyet~~ eyken, _____ (olmak) daha az stresli. (3) (olma) çok
(satın alırsanız), ben _____ (olsun,ım). (1) _____ (gidiyoruz). (2)
- 11) Lucy işini yakında _____ (bırakmazsa) _____ (çıldırır) demektir. (1)

2.4. Şartları Yeniden Yazma. Bu fikirleri koşullu yapılar kullanarak nasıl ifade edebilirsiniz?

Model: Avustralya'ya gidip gitmeyeceğimi bilmiyorum. Diyelim ki gidiyorum.
Orada ne yapabilirim?

- Avustralya'ya gitseydim Sidney Liman Köprüsü'nü görebilirdim.

- 1) Derse gittin. Onu gördüğüm üne eminim. Onu görmekten kaçmış olamazsun.
- 2) İngilizce konuşma konusunda nasıl deneyim kazanabilirim?

- 3) Umarım yarıyıl sonu sınavlarında başarısız olmam. olduğumdan nasıl emin olabilirim geçmek?
- 4) Çevrimiçi bir testi her bitirdiğimizde, bir Kahve.
- 5) Güneş parladığında her zaman daha mutlu hissediyorum.
- 6) Keşke kütüphane şimdi açık olsaydı. Bir kitap ödünç almak istiyorum.

3. Tartışma

3.1. Oyun motoru kavramını kendi kelimelerinizle açıklayın

3.2. Bahsedilen oyun motorlarından birini seçin ve "Bu oyun motorunun avantajları ve dezavantajları nelerdir" konulu bir rapor hazırlayın.

Unity, Unreal Engine, GameMaker, CryEngine, MonoGame, Construct

4. Ek Okuma

4.1. Aşağıdaki metni okuyunuz ve tercüme ediniz. Eğer bir sözlük kullanabilirsiniz gerekli

Yapay zeka

Yapay zeka, insan zekası süreçlerinin makineler, özellikle bilgisayar sistemleri tarafından simülasyonudur. Yapay zekanın özel uygulamaları arasında uzman sistemler, doğaçlama, dil işleme, konuşma tanıma ve makine vizyonu yer almaktır.

Yapay zeka etrafındaki hype hızlanırken, satıcılar ürünlerinin ve hizmetlerinin yapay zekayı nasıl kullandığını tanıtmak için çabalıyorlar. Genellikle AI olarak adlandırdıkları şey, makine öğrenimi gibi AI'nın yalnızca bir bileşenidir. Yapay zeka, makine öğrenimi algoritmaları yazmak ve eğitmek için özel bir donanım ve yazılım temeli gerektirir. Hiçbir programlama dili AI ile eş anlamlı değildir, ancak Python, R ve Java dahil olmak üzere birkaç popülerdir.

Genel olarak, yapay zeka sistemleri, büyük miktarda etiketlenmiş eğitim verisi kullanarak, verileri korelasyonlar ve kalıplar için analiz ederek ve bu kalıpları gelecekteki durumlar hakkında tahminlerde bulunmak için kullanarak çalışır. Bu şekilde, metin sohbetlerinden beslenen bir chatbot, gerçekçi sohbetler üretmeyi öğrenebilir.

insanlarla alışverişe veya bir görüntü tanıma aracı, milyonlarca örneğ i inceleyerek görüntülerdeki nesneleri tanımlamayı ve tanımlamayı ö ğ renebilir.

AI programlama üç bilişsel beceriye odaklanır: ö ğ renme, akıl yürütme ve kendini düzeltme.

Ö ğ renme süreçleri. AI programlamanın bu yönü, veri toplamaya ve verilerin nasıl eyleme geçirilebilir bilgilere dönüştürüleceğ i ilişkin kurallar oluşturmaya odaklanır. Algoritmalar olarak adlandırılan kurallar, bilgi işlem cihazlarına belirli bir görevi nasıl tamamlayacaklarına ilişkin adım adım talimatlar sağ lar.

Akıl yürütme süreçleri. AI programlamanın bu yönü, istenen sonuca ulaşmak iç in doğ ru algoritmayı seçmeye odaklanır.

Kendi kendini düzeltme süreçleri. AI programlamanın bu yönü, algoritmala sürekli olarak ince ayar yapmak ve mümkün olan en doğ ru sonuçları vermelerini sağ lamak iç in tasarlanmıştır.

Yapay zeka, işletmelere operasyonları hakkında daha önce farkında olmayabilecekleri iç gorüler sunabileceğ i ve bazı durumlarda görevleri insanlardan daha iyi yerine getirebileceğ i için önemlidir. Özellikle, ilgili alanların uygun şekilde doldurulduğ undan emin olmak iç in çok sayıda yasal belgeyi analiz etmek gibi tekrarlayan, detay odaklı görevler söz konusu olduğ unda, AI araçları genellikle işleri hızlı ve nispeten az hatayla tamamlar.

Bu, verimlilikte bir patlamayı körüklemeye yardımcı oldu ve bazı büyük işletmeler iç in tamamen yeni iş fırsatlarının kapısını açtı. Mevcut AI dalgasından önce, sürücüler taksilere bağ lamak iç in bilgisayar yazılımı kullanmayı hayal etmek zordu, ancak bugün Uber bunu yaparak dünyanın en büyük şirketlerinden biri haline geldi. İnsanların belirli alanlarda ne zaman ata binmeye ihtiyaç duyacağı inı tahmin etmek iç in gelişmiş makine ö ğ renimi algoritmaları kullanır, bu da sürücülerin ihtiyaç duymadan önce proaktif olarak yola çıkışmasına yardımcı olur. Başka bir örnek olarak, Google, insanların hizmetlerini nasıl kullandığ inı anlamak iç in makine ö ğ renimini kullanarak ve ardından bunları geliştirerek bir dizi ç evrimiç i hizmet iç in en büyük oyunculardan biri haline geldi. 2017'de şirketin CEO'su Sundar Pichai, Google'in "önce AI" şirketi olarak faaliyet gö stereceğ inı açıkladı.

Günümüzün en büyük ve en başarılı işletmeleri, yapay zekayı operasyonlarını geliştirmek ve rakiplerine karşı avantaj elde etmek.

Yapay sinir ağları ve derin ö ğ renme yapay zeka teknolojileri, öncelikle yapay zekanın büyük miktarda veriyi çok daha hızlı işlemesi ve tahminleri insan tarafından mümkün olandan daha doğ ru bir şekilde yapması nedeniyle hızla gelişiyor.

Günlük olarak oluşturulan devasa veri hacmi bir insan araştırmacıyı gömecek olsa da, makine öğrenimi kullanan yapay zeka uygulamaları bu verileri alıp hızla eyleme dönüştürülebilir bilgilere dönüştürebilir. Bu yazı itibariyle, AI kullanmanın birincil dezavantajı, AI programlamanın gerektirdiği büyük miktarda veriyi işlemenin pahalı olmasıdır.

Avantajlar

Detay odaklı işlerde iyi; Veri ağ ırlıklı görevler için azaltılmış süre; Tutarlı sonuçlar sağlar; ve Yapay zeka destekli sanal araçlar her zaman kullanılabilir.

Dezavantajları

Pahalı; Derin teknik uzmanlık gerektirir; Yapay zeka araçları oluşturmak için sınırlı kalifiye işçi temini; Sadece ne gösterildiğiini bilir; ve Bir görevden diğerine genelleme yapma yeteneğinin olmaması. (3500) [https://www.techtarget.com/searchenterpriseai/definition/AI-Yapay Zeka]

4.2. Aşağıdaki soruları cevaplayın:

- 1) Yapay zeka nasıl çalışır?
- 2) Yapay zeka neden önemlidir?
- 3) Yapayın avantajları ve dezavantajları nelerdir? istihbarat?

4.3. Aşağıdaki cümleleri çevirin

- 1) Oyun motorları, aşağıda listelenen sağlayan geliştirme araçları sağlar: basitleştirmek için programcılar tarafından kullanılabilir iş. Kısacası, araçlar sağlayın ve oyun geliştirme için işlevsellik.
- 2) HTML5 motorları geliştiriciler arasında popülerdir oyunlar. Böyle bir Turblenz, için açık bir platform oyun geliştiricileri.
- 3) Bir oyunu geliştirmek, entegre etmek ve para kazanmak için ihtiyaç duyduğumuz tüm temel özellikleri içerir. Haric Ayrıca, kullanımında herhangi bir kısıtlama yoktur, çünkü MIT lisansı altında mevcuttur.

4) "Oyun motoru" terimi, 1990'ların ortalarında,

benzeri birinci şahıs nişancı bilgisayar oyunları

Doom zamanında popülerdi. 5)

Çoğu oyun motoru, belirli bir oyunu belirli bir platformda çalışırmak üzere tasarlanmış ve yapılandırılmıştır. 6) Ve en genelleştirilmiş çok platformlu motorlar bile

Birinci şahıs nişancı oyunları veya yarış oyunları gibi belirli bir türden oyunlar oluşturmak için uygundur. 7) Bu bağlamda oyun daha doğrudan denilebilir.

için kullanılmadığıında motor yetersiz hale gelir.

tasarlandığı oyun veya platform. 8) Bu etki, yazılımın

bunlara dayanan bir dizi uzlaşmadır

oyunun ne olması gerektiğiini tahmin et.

EK OKUMA İÇİN METİNLER

METİN 1. Eğitimin odaklı diller

1. Eğitimin odaklı dillerle ilgili metni okuyun ve çevirin

BASIC

BASIC (yeni başlayanlar için çok amaçlı sembolik talimat kodu), 1960'ların ortalarında John Kemeny ve Thomas Kurtz tarafından Dartmouth Koleji'nde tasarlandı. Acemiler, özellikle bilgisayar bilimleri alanında uzman olmayanlar tarafından kolay öğrenilmesi ve birçok okuyucuya zaman paylaşımı bir bilgisayarda iyi çalımı amaçlanmıştır. Basit veri yapıları ve notasyonu vardı ve yorumlandı: Bir BASIC programı satır satır çevrildi ve çevrildiği gibi yürütüldü, bu da programlamayı bulmayı kolaylaştırdı hatalar.

Küçük boyutu ve basitliği, BASIC'i ilk kişisel bilgisayarlar için popüler bir dil haline getirdi. Son biçimleri, diğer çağdaş dillerin veri ve denetim yapılarının çoğuunu benimsemiştir, bu da onu daha güçlü ancak yeni başlayanlar için daha az kullanışlı hale getirir.

Pascal

1970 Hakkında İsviçre'den Niklaus Wirth, Pascal'ı GOTO ifadeleri olmadan koşullu ve döngü kontrol yapılarının düzenli kullanımını vurgulayan yapılandırılmış programlamayı öğrenmek için tasarladı. Pascal, gösterimde ALGOL'a benzemesine rağmen, karmaşık bilgileri düzenlemek için veri türlerini tanımlama yeteneği sağladı; bu, ALGOL'un yanı sıra FORTRAN ve COBOL'un yeteneklerinin ötesinde bir özellik. Kullanıcı tanımlı veri türleri, programının karmaşık veriler için adlar vermesini sağladı; bu adlar, dil çevirmeninin bir programı çalışmadan önce doğrudan kullanım için kontrol edebileceğini.

1970'lerin sonlarında ve 80'lerde Pascal, programlama eğitimi için en yaygın kullanılan dillerden biriydi. Neredeyse tüm bilgisayarlarda mevcuttu ve aşınılığı, açıklığı ve güvenliğini nedeniyle eğitimin içindir. U kadar üretim yazılımı için de kullanılıyordu.

Logo

Logo, eğitimin basitleştirilmiş

geç

1960'lar

olarak

a

LISP lehçesini oluşturdu; Seymour Papert ve diğerleri bunu MIT'de okul çocukların matematiksel düşünmeyi öğrenmek için kullandılar. LISP'den daha geleneksel bir sözdizimine sahipti ve bilgisayar grafikleri oluşturmak için basit bir yöntem olan "kaplumbağalı grafikleri" içeriyordu. (Adı erkenden geldi).

Kaplumbağ a benzeri bir robotu programlamak için bir proje.) Kaplumbağ a grafikleri, bir nesnenin ekranın etrafında "sol 90" ve "ileri" gibi komutlarla hareket ettirildiği i vücut merkezli talimatlar kullandı. sabit bir çerçeveye açısından değil il, nesne. Özyinelemeli rutinlerle birlikte bu teknik, karmaşık ve çekici kalıpları programlamayı kolaylaştırdı.

Hypertalk

Hypertalk, Apple'ın Macintosh'u için Bill Atkinson tarafından "geri kalanımız için programlama" olarak tasarlandı. İngilizceye benzer basit bir sözdizimi kullanan Hypertalk, herkesin metin, grafik ve sesi program tarafından sağlanan standart düğmeler üzerinde bir fare ile tıklanarak gezinilebilecek "bağlı yığınlar" halinde hızlı bir şekilde birleştirmesini sağladı. Hypertalk, 1980'lerde ve 90'ların başında, sınıf multimedya sunumları için eğitimciler arasında özellikle popülerdi. Hypertalk, nesne yönelimli dillerin birçok özelliğine sahip olsa da (bir sonraki bölümde açıklanmıştır), Apple onu diğer bilgisayar platformları için geliştirmemiş ve yokmasına izin verdi; 1990'larda Apple'ın pazar payı azaldıkça, multimedya görüntülemenin yeni bir çapraz platform yolu Hypertalk'i neredeyse geçersiz kıldı (World Wide Web görüntüleme dilleri bölümüne bakın)

2. Metinle ilgili beş soru sorun

METİ N 2

1. Metni okuyun ve çevirin ve bunun için başlığınızı önerin

Nesneye yönelik analiz ve tasarım (OOAD), bir sistemi etkileşimli nesneler grubu olarak modelleyen bir yazılım mühendisliği yaklaşımıdır. Her nesne, modellenen sistemde ilgili bazı varlıkları temsil eder ve sınıfı, durumu (veri öğeleri) ve davranışları ile karakterize edilir.

Bu birlikte çalışan nesnelerin statik yapısını, dinamik davranışını ve çalışma zamanı dağılımını göstermek için çeşitli modeller oluşturulabilir. Bu modelleri temsil etmek için Birleşik Modelleme Dili (UML) gibi bir dizi farklı gösterim vardır.

Nesneye yönelik analiz (OOA), bir sistemin işlevsel gereksinimlerini analiz etmek için nesne modelleme tekniklerini uygular. Nesneye yönelik tasarım

(OOD), uygulama spesifikasyonlarını üretmek için analiz modellerini detailandırır. OOA sistemin ne yaptığı İna, OOD ise sistemin bunu nasıl yaptığı İna odaklanır.

Nesne yönelimi bir sistem nesnelerden oluşur. Sistemin davranışları, bu nesnelerin işbirliği inden kaynaklanır. Nesneler arasındaki işbirliği i, birbirlerine mesaj göndermeyi iç erir. Bir mesaj göndermek, bir fonksiyonu çağırımaktan farklıdır, çünkü bir hedef nesne bir mesaj aldığı İnda, o mesaja hizmet etmek için hangi fonksiyonun uygulanacağı İna kendisi karar verir. Aynı mesaj, hedef nesnenin durumuna bağlı olarak seçilen birçok farklı işlev tarafından uygulanabilir.

"Mesaj gönderme"nin uygulanması, modellenen sistemin mimarisine ve iletişim kurulan nesnelerin konumuna bağlı olarak değil işir.

Nesneye yönelik analiz (OOA), analiz edilen alanda var olan bilginin kavramsal bir modelini üretmek amacıyla problem alanına bakar. Analiz modelleri, eşzamanlılık, dağitim, kalıcılık veya sistemin nasıl oluşturulacağı gibi mevcut olabilecek herhangi bir uygulama kısıtlamasını dikkate almaz. Uygulama kısıtlamaları, nesne yönelimi tasarım (OOD) sırasında ele alınır. Tasarımdan önce analiz yapılır.

Analizin kaynakları yazılı bir gereksinim beyanı, resmi bir vizyon belgesi, paydaşlarla veya diğer ilgili taraflarla yapılan görüşmeler olabilir. Bir sistem, her biri ayrı ayrı analiz edilen farklı iş, teknolojik veya diğer ilgi alanlarını temsil eden birden çok alana böülünebilir.

Nesne yönelimi analizin sonucu, sistemin işlevsel olarak ne yapması gerekiğinin kavramsal bir model biçiminde bir açıklamasıdır. Bu, tipik olarak bir dizi kullanım durumu, bir veya daha fazla UML sınıf diyagramı ve bir dizi etkileşim diyagramı olarak sunulacaktır. Ayrıca bir tür kullanıcı arayüzü maketi içerebilir. Nesne yönelimi analizin amacı, müşteri tanımlı bir dizi gereksinimleri karşılamaya çalışırken bilgisayar yazılımını tanımlayan bir model geliştirmektir.

Nesneye yönelik tasarım (OOD), seçilen mimarinin dayattığı kısıtlamaları ve işlem hacmi, yanıt süresi, çalışma süresi gibi işlevsel olmayan – teknolojik veya çevresel – kısıtlamaları hesaba katmak için nesne yönelimi analizde üretilen kavramsal modeli dönüştürür. zaman platformu, geliştirme ortamı veya programlama dili.

Analiz modelindeki kavramlar, uygulama sınıfları ve arayüzleri ile eşleştirilir. Sonuç, çözüm alanının bir modeli, sistemin nasıl kurulacağıının ayrıntılı bir açıklamasıdır. (3200)

[[https://www.brainkart.com/article/What-is-OOAD\(Nesneye-yönelik-analiz-ve-tasarım\)-_9969/](https://www.brainkart.com/article/What-is-OOAD(Nesneye-yönelik-analiz-ve-tasarım)-_9969/)]

2. İ ki dakikalık bir konuşma yapın: a) nesne yönelimli analiz, b) nesne yönelimli tasarım

METİ N 3. Render ve Özellikleri

1. Oluşturma veya görüntü sentezi ile ilgili aşağıdaki metni okuyun ve çevirin

Rendering veya görüntü sentezi, bir bilgisayar programı aracılığıyla 2D veya 3D modellen fotogerçekçi veya fotogerçekçi olmayan bir görüntü oluşturma işlemidir. Ortaya çıkan görüntü, render olarak adlandırılır. Kesin olarak tanımlanmış bir dilde veya veri yapısında nesneleri içeren bir sahne dosyasında birden fazla model tanımlanabilir. Sahne dosyası, sanal sahneyi açıklayan geometri, bakış açısı, doku, aydınlatma ve gölgeleme bilgilerini içerir. Sahne dosyasında bulunan veriler daha sonra işlenmek üzere bir işleme programına geçirilir ve bir dijital görüntü veya raster grafik görüntü dosyasına çıkarılır. "Oluşturma" terimi, bir sanatçının bir sahne izlenimi kavramına benzer. "Oluşturma" terimi, aynı zamanda, bir video düzenleme programında, nihai video çıktısını üretmek için efektlerin hesaplanması sürecini tanımlamak için de kullanılır.

Rendering, 3D bilgisayar grafiklerinin ana alt konularından biridir ve pratikte her zaman diğer erleriyle bağlılığındır. Modellere ve animasyona son görüntülerini veren grafik işlem hattındaki son büyük adımındır. 1970'lerden bu yana bilgisayar grafiklerinin artan karmaşaklılığı ile daha belirgin bir konu haline geldi.

Oluşturma, mimaride, video oyunlarında, simülatörlerde, film ve TV görsel efektlerinde ve tasarım görselleştirmesinde her biri farklı özellik ve teknik dengesi kullanan kullanımlara sahiptir. Kullanım için çok çeşitli oluşturucular mevcuttur. Bazıları daha büyük modelleme ve animasyon paketlerine entegre edilmiştir, bazıları bağımsızdır ve bazıları ücretsiz açık kaynaklı projelerdir. İçeride, bir oluşturucu, ışık fiziği, görsel algı, matematik ve yazılım geliştirme dahil olmak üzere birden çok disipline dayalı, özenle tasarlanmış bir programdır.

Oluşturma yöntemlerinin teknik ayrıntıları farklılık gösterse de, bir sahne dosyasında saklanan bir 3B temsilden bir ekranada 2B bir görüntü üretmenin üstesinden gelinmesi gereken genel zorluklar, GPU gibi bir oluşturma aygıtlındaki grafik ardışık düzen tarafından gerçekleştirilir. GPU, karmaşık işleme hesaplamaları gerçek ekleştirmede CPU'ya yardımcı olan amaca yönelik olarak oluşturulmuş bir cihazdır. Bir sahnenin sanal ışık altında nispeten gerçekçi ve tahmin edilebilir görünmesi gerekiyorsa, işleme yazılımının işleme denklemini özmesi gereklidir. Oluşturma denklemi tüm aydınlatma olaylarını hesaba katmaz, bunun yerine bilgisayar tarafından oluşturulan görüntüler için genel bir aydınlatma modeli görevi görür.

3D grafikler söz konusu olduğunda, sahneler önceden oluşturulabilir veya gerçek zamanlı olarak oluşturulabilir. Ön işleme, sahnelerin önceden oluşturulabildiği, genellikle film oluşturma için kullanılan yavaş, hesaplama açısından yoğun bir işlemidir; gerçek zamanlı oluşturma ise genellikle dinamik olarak sahneler oluşturulması gereken 3D video oyunları ve diğer uygulamalar için yapılır. 3B donanım hızlandırıcıları, gerçek zamanlı işleme performansını iyileştirebilir. (2700) [[https://en.wikipedia.org/wiki/Rendering_\(computer_graphics\)](https://en.wikipedia.org/wiki/Rendering_(computer_graphics))]

2. Render yapmanın ana fikri nedir?

3. Plana göre metnin kısa bir özetini yazın (tüm ifadeleri kullanmak zorunda olmadığıınızı unutmayın).

Açıklama planı: 1)

Makalenin başlığı (metin): -

Metnin adı - Okuduğum

makalenin başlığı ... - Makalenin başlığı,

makaleye atıfta bulunduğuunu belirtir ... 2) makalenin yazarı, nerede ve

ne zaman yayınlandı: - Makalenin yazıldığı ... ve yayınlandı - Ne yazık

ki yazarın adı bilinmiyor. - Çevrimiçi bir kaynaktan alınan metin. 3)

makalenin ana fikri (metin): - Metnin ana fikri şudur ... - Makale soruna

ayrılmıştır ... - Makalenin amacı ... 4) makalenin içeriği, gerçekler,

başlıklar, şekiller: - Yazar, materyalin sunumuna ...

- Yazar konuya okuyucunun dikkatini çeker... - Özellikle dikkat edilir... - Makalede sunulan verilere göre... - Vurgulanır (belirtilir/ doğ rulanır/ ispatlanır/ yalanlanır/ sorgulanır).... - Yazar ayrıntılı bir analiz yapar... - Ayrıca, gerçekler açıklanır... (örnekler verilir.../vakalar açıklanır...). - Sonuç olarak, yazar notlar (iddia eder / vurgular, odaklanır ..) - Özetle, yazar sonuca varır... 5) sizin fikriniz: - Makale bana ilginç (bilgilendirici) geldi, çünkü içinde

verilir ... -

Hayattan bazı örnekler vermeyi gereklili görüyorum (diğer makaleler) bu makaledekiyle aynı fenomeni açıklar.
Metin.

- Yazar tarafından önerilen fikirler bu şekilde uygulanabilir.
gibi alanlar - ...

Bazı hükümler çelişkili görünüyor çünkü.... - Makale okuyanlar için yararlı olabilir (... alanında çalışmak). ...

METİ N 4. Yapay Zekanın Dört Türü

1. Yapay zeka türleri ile ilgili aşağıdaki metni okuyup tercüme edin ve bir bunun kısa özeti. Yukarıda verilen planı takip edin

Reaktif Makineler

Reaktif bir makine, en temel AI

ilkelerini takip eder ve adından da anlaşılacağı gibi, zekasını yalnızca önündeki dünyayı algılamak ve tepki vermek için kullanabilir. Reaktif bir makine hafıza depolayamaz ve sonuç olarak gerçek zamanlı olarak karar vermemi bilgilendirmek için geçmiş deneyimlere güvenemez.



Dünyayı doğrudan algılamak, reaktif makinelerin yalnızca sınırlı sayıda özel görevi tamamlamak üzere tasarlandığı anlamına gelir. Bununla birlikte, reaktif bir makinenin dünya görüşünü kasıtlı olarak daraltmak, herhangi bir malihet düşürücü önlem değil ve bunun yerine, bu tür AI'nın daha güvenilir ve güvenilir olacağı anlamına gelir - her seferinde aynı uyararlara aynı şekilde tepki verir.

Reaktif makinenin ünlü bir örneği, 1990'larda IBM tarafından satranç oynayan bir süper bilgisayar olarak tasarlanan ve uluslararası büyük usta Gary Kasparov'u bir oyunda mağlup eden Deep Blue'dur. Deep Blue sadece bir satranç tahtasındaki taşları tanımlayabiliyor ve her bir hamlenin satranç kurallarına göre nasıl yapıldığıını biliyor, her bir taşın mevcut konumunu kabul ediyor ve o anda en mantıklı hamlenin ne olacağıını belirleyebiliyordu. Bilgisayar, rakibinin gelecekteki olası hamlelerini takip etmiyor veya kendi taşlarını daha iyi bir konuma getirmeye çalışmıyordu. Her dönüş, önceden yapılmış diğer hareketlerden ayrı olarak, kendi gerçek ekiliğine olarak görülmüyordu.

Oyun oynayan bir reaktif makinenin başka bir örneği ise Google'ın AlphaGo'sudur. AlphaGo ayrıca gelecekteki hamleleri değil erlendirmekten acizdir, ancak mevcut oyunun gelişmelerini değil erlendirmek için kendi sinir ağına güvenir ve daha karmaşık bir oyunda Deep Blue'ya karşı bir avantaj sağlıyor. AlphaGo ayrıca 2016'da şampiyon Go oyuncusu Lee Sedol'u yenerek oyunun dünya çapındaki rakiplerini de geride bıraktı.

Kapsamlı sınırlı ve kolayca değil işitiremese de, reaktif makine yapay zekası bir karmaşıklık düzeyine ulaşabilir ve tekrarlanabilir görevleri yerine getirmek için oluşturulduğunda güvenilirlik sunar.

Sınırlı Bellek Sınırlı

Bellek yapay zekası, bilgi toplarken ve potansiyel kararları tartarken önceki verileri ve tahminleri saklama yeteneğini sahiptir - esasen bir sonraki adımda ne olabileceğini dair ipuçları için geçmişe bakar. Sınırlı bellekli yapay zeka, reaktif makinelerden daha karmaşıktır ve daha büyük olanaklar sunar.

Bir ekip sürekli olarak bir modeli yeni verilerin nasıl analiz edileceği ve kullanılacağı konusunda eğitildiğiinde veya modellerin otomatik olarak eğitilip yenilenebilmesi için bir AI ortamı oluşturulduğunda sınırlı bellek AI oluşturulur. Makine öğreniminde sınırlı bellekli yapay zeka kullanılırken altı adım izlenmelidir: Eğitilen verileri oluşturulmalı, makine öğrenmesi modeli oluşturulmalı, model tahminlerde bulunabilmeli, model oluşturulmalıdır.

insan veya çevresel geri bildirim alabilen, bu geri bildirimin veri olarak saklanması ve bu adımların bir döngü olarak tekrarlanması gereklidir.

Zihin Teorisi Zihin

Teorisi tam da budur – teorik. Bu bir sonraki yapay zeka seviyesine ulaşmak için gerekli teknolojik ve bilimsel yeteneklere henüz ulaşılamadık.

Kavram, diğer canlıların kendi davranışlarını etkileyen düşünce ve duygulara sahip olduğumu anlamaının psikolojik öncülüğe dayanmaktadır. Yapay zeka makineleri açısından bu, yapay zekanın insanların, hayvanların ve diğer makinelerin nasıl hissettiğini anlayabileceğini ve öz-yansıtma ve kararlılık yoluyla karar verebileceğini ve daha sonra bu bilgiyi kendi kararlarını vermek için kullanacağı anlamına gelir. Esasen, makinelerin "zihin" kavramını, karar vermedeki duyguların dalgalanmalarını ve diğer psikolojik kavamları gerçek zamanlı olarak kavrayabilmeleri ve işleyebilmeleri, insanlar ve yapay zeka arasında iki yönlü bir ilişki yaratıbilmeleri gereklidir.

Öz-farkındalık

Yapay zekada Zihin Teorisi bir kez oluşturulduğunda, gelecekte çok iyi bir zamanda, son adım AI'nın kendinin farkında olması olacaktır. Bu tür bir yapay zeka, insan düzeyinde bir bilinc sahiptir ve dünyadaki varlığıının yanı sıra başkalarının varlığını ve duygusal durumunu da anlar. Sadece onlara ne ilettilerine değil, nasıl ilettilerine dayanarak başkalarının neye ihtiyaç duyabileceğini anlayabilecektir.

Yapay zekada öz-farkındalık, hem insan araştırmacılarının bilincin öncülünü anamasına hem de makinelere yerleştirilebilmesi için bunu nasıl kopyalayacağıını öğrenemeye dayanır. (4300)

[<https://www.govtech.com/computing/understanding-the-four-types-of-suni-intelligence.html>]



2. Aşağıdaki soruları yanıtlayın: AI neden önemlidir? yapay zeka nasıl iş?

3. Yapay Zekanın kısa bir tarihi hakkında bir rapor hazırlayın

METİ N 5. Linux'un Temel Kavramları

1. Linux sistemi hakkındaki metni okuyun ve çevirin

Linux, diğer UNIX sistemlerine çok benziyor ve hissediyor; gerçekten de, UNIX uyumluluğu, Linux projesinin ana tasarım hedefi olmuştur.

Ancak Linux, çoğu UNIX sisteminden çok daha gençtir. Gelişimi 1991'de Finlandiyalı bir üniversite öğrencisi olan Linus Torvalds, Intel'in PC uyumlu CPU serisindeki ilk gerçek 32 bit işlemci olan 80386 işlemci için küçük ama bağımsız bir çekirdek geliştirmeye başladığında başladı. rasgele dosya sayısı (ancak yalnızca salt okunur bellek eşlemesi 1.0'da uygulandı).

Bu sürüme bir dizi ekstra donanım desteği dahil edilmiştir.

Hala Intel PC platformuyla sınırlı olmasına rağmen, donanım desteği, disket ve CD-ROM aygıtlarının yanı sıra ses kartları, çeşitli fareler ve uluslararası klavyeleri içerecek şekilde büyümüştü. 80387 matematik yardımcı işlemcisi olmayan 80386 kullanıcı için çekirdekte kayan nokta öykünmesi sağlandı. Paylaşılan bellek, semaforlar ve mesaj kuyrukları dahil olmak üzere System V UNIX tarzı süreçler arası iletişim (IPC) uygulandı.

Linux çekirdeği tamamen özellikle Linux projesi için sıfırdan yazılmış kodlardan oluşur, Linux sistemini oluşturan destekleyici yazılımların çoğu Linux'a özel değil, ancak bir dizi UNIX benzeri işletim sisteminde ortaktır. Özellikle Linux, Berkeley'in BSD işletim sisteminin, MIT'nin X Pencere Sisteminin ve Özgür Yazılım Vakfı'nın GNU projesinin bir parçası olarak geliştirilen birçok aracı kullanır.

Araçların bu paylaşımı her iki yönde de işe yaradı. Linux'un ana sistem kitaplıkları, GNU projesi tarafından oluşturuldu, ancak Linux topluluğu, eksiklikleri, verimsizlikleri ve hataları ele alarak kitaplıkları büyük ölçüde geliştirdi. GNU C derleyicisi (gcc) gibi diğer bileşenler, doğrudan Linux'ta kullanılmak için zaten yeterince yüksek kalitedeydi. Linux altındaki ağ yönetim araçları aşağıda kılınmıştır:

kod ilk olarak 4.3 BSD için geliştirildi, ancak FreeBSD gibi daha yeni BSD türevleri karşılığında Linux'tan kod ödünç aldı. Bu paylaşımı örnek olarak Intel kayan nokta öykünmesi matematik kitaplığı ve PC ses donanımı aygıt sürücülerini dahildir.

Bir bütün olarak Linux sistemi, belirli bileşenlerin bütünlüğünü koruma sorumluluğu una sahip küçük gruplar veya bireylerle İnternet üzerinden işbirliği yapan gevşek bir geliştiriciler ağı tarafından sürdürülür.

Az sayıda genel İnternet dosya aktarım protokolü (FTP) arşiv sitesi, bu bileşenler için fiili standart depolar olarak işlev görür.

Dosya Sistemi Hiyerarşi Standardı belgesi, çeşitli sistem bileşenleri arasında uyumluluğu sağlamanın bir yolu olarak Linux topluluğu tarafından da korunur.

Bu standart, standart bir Linux dosya sisteminin genel düzenini belirtir; yapılandırma dosyalarının, kitaplıkların, sistem ikili dosyalarının ve çalışma zamanı veri dosyalarının hangi dizin adları altında saklanması gerekiğini belirler.

Linux Dağıtımları

Teorik olarak, herkes gerekli sistem bileşenlerinin en son revizyonlarını FTP sitelerinden alıp derleyerek bir Linux sistemi kurabilir. Linux'un ilk günlerinde, bir Linux kullanıcısının yapması gereken tam olarak buydu. Ancak Linux olgunlaşıkça, çeşitli kişiler ve gruplar, kolay kurulum için standart, önceden derlenmiş paket setleri sağlayarak bu işi daha az zahmetli hale getirmeye çalıştı.

Bu koleksiyonlar veya dağıtımlar, temel Linux sisteminden çok daha fazlasını içerir. Bunlar genellikle ekstra sistem kurulum ve yönetim yardımcı programlarının yanı sıra haber sunucuları, web tarayıcıları, metin işleme ve düzenleme araçları ve hatta oyunlar gibi yaygın UNIX araçlarının birçoğu unun önceden derlenmiş ve kuruluma hazır paketlerini içerir.

İlk dağıtımlar, tüm dosyaları uygun yerlere açmanın bir yolunu sağlayarak bu paketleri yönetti. Ancak modern dağıtımların önemli katkılarından biri gelişmiş paket yönetimidir. Günümüzün Linux dağıtımları, paketlerin zahmetszizce kurulmasına, yükseltilmesine veya kaldırılmasına izin veren bir paket izleme veritabanı içerir. (3900)

2. Kendinizi test edin. Bazı soruların cevaplarını bilmiyorsanız, daha sonra ödev için internette bulabilirsiniz.

Linux hakkında ne biliyorsun?

1. Linux geliştirmesi ... ile mi başladı?

- a) 1978
- b) 1991
- c) 2001

2. Linux'un kaç temel öğesi veya bileşeni vardır?

- a) 5
- b) 3
- c) 6

3. Linux işletim sisteminin ana bileşeni nedir? a)

donanım b) çekirdek c) kabuk 4. Linux kabuğu nedir?

a) kullanıcı ve çekirdek arasında mevcut bir kullanıcı arayüzü. b) bir dizi farklı durumda olabilen bir süreç türü. c) donanım ve donanım arasında köprü görevi gören bir kaynak yöneticisi yazılım.

5) CLI nedir? a)

genellikle metni yürütülecek girdi olarak kabul eden bir komut satırı programı veya işletim sisteminin işlevlerini çalıştırın.

b) Kullanıcıların elektronik cihazlarla grafik simgeler ve görsel göstergeler aracılığıyla etkileşime girmesine olanak tanıyan bir insan-bilgisayar arayüzü.

6. Linux sistemi bir bütün olarak korunur...

- a) uzun vadeli kamu desteği ile. b)

İnternet üzerinden işbirliği yapan gevşek bir geliştirici ağının tarafından.

7. Dosyaları kaldırmak için kullanılan ad komutu. a)

- sil b) dr c) rm d) Yukarıdakilerin hiçbiri

8. Linux'u geliştiren kişinin adını söyleyin. a)

Linus Torvalds b) Dennis Ritchie

c) Prof. Andrew S. Tannenbaum d)

Yukarıdakilerin hiç biri

9. Linux'ta dosya adının maksimum boyutu (bayt cinsinden) ne olabilir?

a) 64 bayt b)

32 bayt c)

128 bayt d)

255 bayt 10.

Linux işletim sisteminin özellikleri nelerdir? a) Açık

kaynaklıdır ve kullanımı ücretsizdir. b) Açık kaynak

değ ildir ve kullanımı ücretsiz değil ildir. c) Çok düşük

donanım gereksinimlerine sahiptir ve ağ oluşturma için güçlü desteği i

kolaylaştırır. d) Yukarıdakilerin hiç biri.

Görevin cevap anahtarı: 1.b); 2.a); 3.b); 4.a); 5.a); 6.b); 7.b); 8.d); 9.c); 10 A).

METİ N 6. Müşterilerle İletişim İçin Bir Geliştirici Kılavuzu

1. Bir yazılım geliştiricisi ile bir müşteri arasındaki başarılı bir konuşma hakkında aşağıda metni okuyun ve tercüme edin

Müşterilerle iletişim kurmakta her zaman kendini rahat hissetmeyen bir geliştiriciyiniz ve tek yapmak istediğiniz sadece kod ve daha fazla kod ise, bu makale akımda sizinle birlikte oluşturuldu.

1) Müşterilere bir görev aldığınızınızı bildirmek önemli olarak

aklımdan geçen ilk şey, müşterilere size atanmış bir görev aldığınızı belirtmektir. Belki bazılarınız bunun hiç de akıllıca olmadığını düşünübilir, ancak bu küçük adım günlük iletişimde çok önemlidir.

Durum A) Müşteri bir görev atar. Bakmak için bir geliştirici mevcuttur.

Çözüm: Müşteriye hemen bakacağınızı ve daha bilinçli bir geri bildirimle kendisine döneceğinizizi bildirmelisiniz veya hızlı bir göz atabilir ve ilk yanıtta ne yapılması gerekiyor ini daha iyi anlamak için bazı sorular sağlayabilirsiniz.

Durum B) Müşteri bir görev atar. Bir geliştirici hemen bakmakta özgür değil ildir.

Çözüm: Görevle ilgili bilgi verdikleri için onlara teşekkür edin ve mümkün olan en kısa sürede başlayacağınızı bildirin. Bu şekilde, şu anda müsait olmadığıınızı, ancak olur olmaz, üzerinde çalışacağınızı bilmelerini sağladınız.

Yapma:

Çözüm süreci için müsait olmadığıınız süre boyunca bir veya birkaç gün görevi görmezden gelin. Bu görevi neden hemen yapamayacağınızı dair ek bilgileri paylaşın. Karmaşık olmaması gereken şeyleri karmaşık hale getireceksiniz.

2) İ stemciyi görevin ilerleyişi hakkında sık sık güncelleme

Çoğu durumda, bir geliştirici müşteriye görevi başladığını bildirir ve bitirmeden herhangi bir güncelleme vermez. Tamamlanması bir veya iki gün gerektiren bir görevden bahsediyorsak, müşteriye çok fazla güncelleme yüklemek istemediği imiz için kesinlikle doğrudu bir yaklaşımındır. Ancak, en az 7- gerektiren daha fazla zaman alan görevler

10 gün yerine getirilmesi için zaman zaman güncellemler verilmelidir.

Durum güncellemleri şunlar

olmamalıdır: Çok büyük veya
çok kısa (Çok sık (yani üzerinde 7 gün çalışıyorsanız, 3 güncelleme
yeterli)

Çözüm: Şimdi onunla başlayabilir ve günün sonunda müşteriyi ilerleme hakkında güncelleyebilirsiniz. En azından bir şeyler yaptın.

Çözüm: Neler olup bittiğini karşı dürüst olmalısınız. Acil durum görev(ler)iyle meşguldünüz ve şimdi bu görevi başlıyorsunuz. Özür dileyecek ve bu görevi öncelik vereceksiniz. Belki müşteri şu anda bu açıklamalardan memnun olmayacağından korkmayı. Bir süre sonra dürüstlüğüünü takdir edecek.

Yapma:

Gerçekte henüz başlamadığınız bir şeyi yaptığınız konusunda müşteriye yalan söyleyin. Unutmayın ki yalanlar her zaman sonunda ortaya çıkar.

3) Çok uzun yazılar yazmaktan kaçınmak

Müşterinin, özellikle kısa ve sorunsuz bir şekilde bir şeyler söylenebiliyorsa, uzun yazıları okumaya vakti olmadığıını hatırlamalısınız. Çok uzun bir yazı yazdışsanız,

görevi iyi anlamadı. Bu yüzden her zaman işleri basitleştirmeye ç alışın, eğ er

Yapabilmek.

4) Müşteriyi tatilleriniz ve yükümlülükleriniz hakkında zamanında bilgilendirmek Müşteriyi, size yönelik görevleri planlayabilmesi için planladığınız tatil hakkında önceden

bilgilendirin. Ayrıca, ilerlemelerinin ortasında gitmiş olacağınızı ve bunları kontrol etmek için müsait olmadığıınızı bilerek karmaşık görevler almayın. Tatile çıkmak üzereyken, acil bir durum olduğunda hizmette olması için başka bir geliştiriciyi adlandırın.

Hastalandığınız veya başa çıkışmanız gereken bazı beklenmedik şeyler olduğunda, ekip arkadaşlarınızdan veya ekip liderinizden, alıştığınız projelerde belirli bir süre müsait olmayacağına dair hızlı bir not yazmasını isteyin.

Bu yaklaşımalarla müşteriden çok fazla saygı göreceksiniz.

Ancak proje müşteri nedeniyle yavaş ilerliyorsa ve şimdiden 2 hafta gecikmeniz varsa ne yapmalısınız? Bu kesinlikle bireysel bir yaklaşım ve durumun daha derin bir analizini gerektiren bir şeydir. Nasıl tepki vereceğiniz konusunda çok kararsızsanız, aşağıdaki soru doğrudan karar vermenize yardımcı olabilir: Bu proje şirketiniz için ne kadar önemlidir? Bu proje gerçekten 2 hafta içinde mi yoksa 4-6 hafta sonra mı başlayacak?

haftalar?

Siz yokken birisi pozisyonunuza yerine getirebilir mi?

Denemeye ve tatilinizi projeye doğrudan kaydılmaya istekliyiniz, müşteriye işleri halletmek için kalmaya karar verdığınızda bildirmeli, ancak ne kadar süre müsait olacağınız konusunda net bir son tarih belirlemelisiniz. Bu şekilde, başka bir son teslim tarihi geçtiğinde kendinizi suçlu hissetmezsiniz.

Günlük işlerinizde bu becerilerde ustalaştıysanız, müşteriler muhtemelen sizinle çalışmaktan zevk alırlar.

Müşteri kızgın ve kaba olduğunda ne yapmalı?

Deneyimlerime göre, bir mesajla duygusal olarak ilgilenebilmek ve soğuk davranışlarda yanıt vermek önemlidir. Mesajın tonu sakin ve saygılı olmalıdır. Hatanız olsa bile korkmayın. Durumu inceleyin – bu neden oldu ve düzeltilebilir mi?

Bir çözüm önermeye çalışın. Bu durumla nasıl başa çıkacağınızdan emin değilseniz, ekip liderinizden araya girmesini isteyin. Gerçek şu ki, iyi iletişim gelecekteki uzun vadeli bir ilişkinin kesin kanıdır. (4190)

[Kısaltılmış itibaren https://inchoo.net/life-at-inchoo/developers-guidecommunication-clients/ Antonija Tadic tarafından]

3. Yukarıda bahsedilen durumlardan birinde olduğunuzu hayal edin.

Sınıfta grup arkadaşlarınızla 'bir müşteriyle iletişim kurma' konusunda bir diyalog'u dramatize edin.

METİ N 7. Polimorfizm, Kapsülleme ve Kalıtım Birlikte Çalışır

1. Aşağıdaki metni okuyun ve çevirin

Düzgün uygulandığında, polimorfizm, kapsülleme ve kalıtım, süreç odaklı modelden çok daha sağlam ve ölçülebilir programların geliştirilmesini destekleyen bir programlama ortamı oluşturmak için birleştir. İyi tasarlanmış bir sınıf hiyerarşisi, geliştirmeye ve test etmeye zaman ve emek harcadığınız kodu yeniden kullanmanın temelidir. Kapsülleme, uygulamalarınızı zaman içinde sınıflarınızın genel arabirimine bağlı olarak kodu bozmadan geçirmenize olanak tanır. Polimorfizm, temiz, mantıklı, okunabilir ve esnek kod oluşturmanıza olanak tanır.

Gerçek dünyadaki iki örnekten otomobil, nesne yönelimli tasarımının gücünü daha eksiksiz bir şekilde gösterir. Köpekleri miras açısından düşünmek eğlenceli, ancak arabalar daha çok programlara benzer. Tüm sürücüler, farklı araç türlerini (alt sınıfları) sürmek için mirasa güvenir.

Araç okul otobüsü, Mercedes sedan, Porsche veya aile minibüsü olsun, sürücüler az çok direksiyonu, frenleri ve gaz pedalını bulup çalıştırabilir. Bir miktar dişli taşlamadan sonra, çoğu insan bir vites değil istirme ile otomatik vites arasındaki farkı bile yonetebilir, çünkü temel olarak ortak üst sınıfları olan şanzımanı anırlar.

Araçlarda her zaman kapsülleme özelliklerle insanlar arayüzü. Fren ve gaz pedalları, ayaklarınızla çalıştırabileceğiniz kadar basit bir arayüzle inanılmaz bir karmaşıklık dizisini gizler! Motorun uygulanması, frenlerin tarzı ve lastiklerin boyutu, pedalların sınıf tanımlıyla nasıl etkileşim kurduğumuz üzerinde hiç bir etkiye sahip değilildir.

Nihai nitelik olan polimorfizm, otomobil üreticilerinin temelde aynı araç üzerinde çok çeşitli seçenekler sunma yeteneklerine açıkça yansır. Örneğin, kilitlenmeyen bir fren sistemi veya geleneksel frenler, hidrolik veya kremayerli direksiyon ve 4, 6 veya 8 silindirli motorlar alabilirsiniz. Her iki durumda da, durmak için yine de fren pedalına basacaksınız,

yön değiştirmek için direksiyon simidi ve hareket etmek istediğiinizde gaza basın. Aynı arayüz, bir dizi farklı uygulamayı kontrol etmek için kullanılabilir.

Gördüğünüz gibi, tek tek parçaların bir araba olarak bilinen nesneye dönüştürülmesi kapsülleme, kalıtım ve polimorfizm uygulamasıyla gerçekleşir. Aynı durum bilgisayar programları için de geçerlidir. Nesne yönelimli ilkelerin uygulanmasıyla, karmaşık bir programın çeşitli bölgümleri, uyumlu, sağlam ve sürdürülebilir bir bütün oluşturmak için bir araya getirilebilir.

Bu bölümün başında belirtildiği gibi, her Java programı nesne yönelimlidir. Ya da daha doğrusu, her Java programı kapsülleme, kalıtım ve polimorfizm içerir. Bu bölümün geri kalanında ve sonraki birkaç bölümde gösterilen kısa örnek programlar bu özelliklerin hepsini göstermeyecek, ancak yine de mevcutturlar. Göreceğiniz gibi, Java tarafından sağlanan özelliklerin çoğu, kapsamlı bir şekilde kapsülleme, kalıtım ve polimorfizmden yararlanan yerleşik sınıf kitaplıklarının bir parçasıdır.

[https://www.brainkart.com/article/Object-Oriented-Programming_10384/]

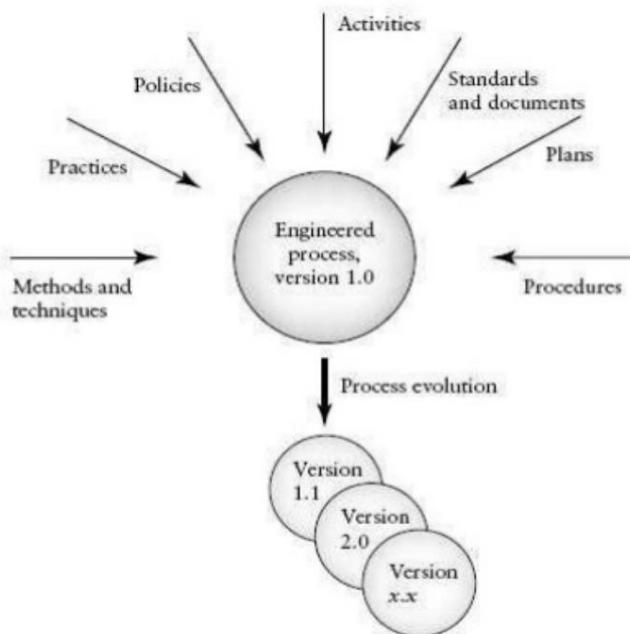
2. Metinde polimorfizm, kapsülleme ve kalıtımıla ilgili beş soru sorun

METİ N 8. Yazılım Kalitesinde Sürecin Rolü

Yüksek kaliteli yazılım ürünlerine duyulan ihtiyaç, meslektaşları kullanılabilirlik, test edilebilirlik, sürdürülebilirlik ve güvenilirlik gibi kalite faktörlerini belirlemeye ve ölçmeye ve bu olumlu niteliklere sahip kaliteli ürünlerin üretimini destekleyen mühendislik uygulamalarını belirlemeye zorlamıştır. Yüksek kaliteli yazılımların geliştirilmesine katkıda bulunan tespit edilen uygulamalar arasında proje planlama, gereksinim yönetimi, resmi özelliklerin geliştirilmesi, bilgi gizleme ve kapsülleme kullanımıyla yapılandırılmış tasarım, tasarım ve kodun yeniden kullanımı, denetimler ve gözden geçirmeler, ürün ve süreç önlemleri, yazılım profesyonellerinin eğitimi ve eğitimi, CASE araçlarının geliştirilmesi ve uygulanması, etkili test tekniklerinin kullanılması ve test faaliyetlerinin tüm yaşam döngüsüne entegrasyonu. Bu bireysel en iyi teknik ve yönetimsel uygulamaları belirlemenin yanı sıra, yazılım araştırmacıları, bunları yüksek kaliteli bir yazılım geliştirme süreci bağlamında bütünlüğünün önemini olduğunu fark ettiler.

Yazılım geliştirmede süreç nedir?

Yazılım mühendisliği alanındaki süreç, yazılım mühendislerinin bir yazılım sistemini ve proje ve test planları gibi ilişkili yapılarını geliştirmek ve sürdürmek için kullandıkları yöntemler, uygulamalar, standartlar, belgeler, faaliyetler, politikalar ve prosedürler kümesidir. tasarım belgeleri, kod ve kılavuzlar.



Şekil 13. Bir mühendislik sürecinin bileşenleri

Ayrıca, mevcut bir yazılım geliştirme sürecine geçici bir şekilde bireysel uygulamalar eklemenin tatmin edici olmadığı da açıktır. Çoklu mühendislik ürünü gibi yazılım geliştirme süreci de tasarlanmalıdır. Yani tasarlanmalı, uygulanmalı, değerlendirilmeli ve sürdürülmelidir. Diğer mühendislik disiplinlerinde olduğu gibi, bir yazılım geliştirme süreci tutarlı ve öngörlülebilir bir şekilde gelişmeli ve en iyi teknik ve yönetimsel uygulamalar sistematik bir şekilde bütünlendirilmelidir. Bu modeller, bir kuruluşun mevcut yazılım sürecini değil etlendirmesine ve durumunun anlaşılmasına olanak tanır. Modeller, tarihsel süreç evrimi ve kalite ilkelerinin uygulanmasıyla tutarlı olarak artan süreç iyileştirmesi için güçlü destek sağlar. Modeller endüstriden büyük ilgi gördü ve birçok başarı kaydedilen süreç iyileştirme çabalarına kaynaklar yatırıldı.

Endüstride geniş kabul görmüş tüm yazılım süreç iyileştirme modelleri, odaklandıkları anlamda üst düzey modellerdir.

tasarım ve test gibi belirli yazılım geliştirme alt süreçlerini değil erlendirmek ve geliştirmek için yeterli desteği sunmamaktadır. Çok sayıda yazılım mühendisi, testin kaliteli bir yazılım sürecinin hayatı bir bileşeni olduğunu ve yazılım geliştirme ve bakımı sırasında gerçekleştirdiğimiz en zorlu ve maliyetli faaliyetlerden biri olduğunu konusunda hemfikirdir. (2900) [https://www.brainkart.com/article/Role-of-process-in-software-quality_9136/]

2. Aşağıdaki 'süreç', 'yazılım geliştirme süreci', 'modeller' ve 'teknikler' terimlerini kendi kelimelerinizle açıklayın.

METİ N 9. Mobil Bilişim Uygulamaları

1. Mobil bilgi işlemle ilgili aşağıdaki metni okuyun ve çevirin

Emlakçılar için

Emlakçılar hem evde hem de sahada çalışabilirler. Mobil bilgisayarlarla daha üretken olabilirler. Müşterileri ile dışarı çıktılarında evden, ofisten veya arabadan yapabilecekleri çoklu listeleyici hizmetlerine erişerek güncel gayrimenkul bilgilerine ulaşabilirler. Başvurular anında yapılabildiğiinden, müşterilere belirli evler veya mahalleler hakkında anında geri bildirim ve daha hızlı kredi onayları sağlayabilirler.

Bu nedenle, mobil bilgisayarlar müşterilere daha fazla zaman ayırmalarına izin verir.

Acil servisler

Acil servislerin dahil olduğu yerlerde hareket halindeyken bilgi alma yeteneği hayatı önem taşır. Bir olayın adresi, türü ve diğer detayları ile ilgili bilgiler, mobil bilgisayarlar kullanılarak bir CDPD sistemi aracılığıyla, olayın yakınında bulunan bir veya birkaç uygun mobil birime hızlı bir şekilde gönderilebilir. Burada CDPD sisteminde uygulanan güvenilirlik ve güvenlik büyük avantaj sağlayacaktır.

Şekil 14. Polis olayı bilgi ekranı

mahkemelerde

Savunma avukatları mobil bilgisayarları mahkemeye verebilir. Rakip avukat aşina olmadığı bir davaya atıfta bulunduğu unda, dava ve ilgili emsaller hakkında bilgi toplayabilecekleri evrimiçi yasal veri tabanı hizmetlerine doğrudan, gerçek zamanlı erişim sağlama için bilgisayarı kullanabilirler. Bu nedenle mobil bilgisayarlar, çok sayıda bilgiye anında erişim sağlayarak insanları daha iyi bilgilendirir ve hazırlar.

şirketlerde

Yöneticiler, örneğin önemli müşterilere yönelik kritik sunumlarda mobil bilgisayarları kullanabilirler. En son pazar payı bilgilerine ulaşabilirler. Küçük bir teneffüste, bu bilgilerden yararlanmak için sunumu gözden geçirebilirler. Olası yeni teklifler hakkında ofisle iletişim kurabilir ve yeni tekliflere verilen yanıtları tartışmak için toplantı çağrısında bulunabilirler. Bu nedenle, mobil bilgisayarlar rekabet avantajlarından yararlanabilir.

Stok Bilgileri Harmanlama/Kontrol

Stoka erişimin çok sınırlı olduğu ortamlarda, yani fabrika depolarında. Bir mobil bilgisayar aracılığıyla erişilen küçük taşınabilir elektronik veritabanlarının kullanılması ideal olacaktır. Derlenen veriler, tüm stok bilgilerini tutan bir CDPD ağ aracılığıyla doğrudan merkezi bir veritabanına yazılabilir, bu nedenle verilerin daha sonraki bir tarihte merkezi bilgisayara aktarılması gerekli değildir. Bu, bir stok sayımının yapıldığıandan itibaren

tamamlandığında, taşınabilir bilgisayarlardaki veri girişi ile merkezi veri tabanı arasında herhangi bir tutarsızlık yoktur.

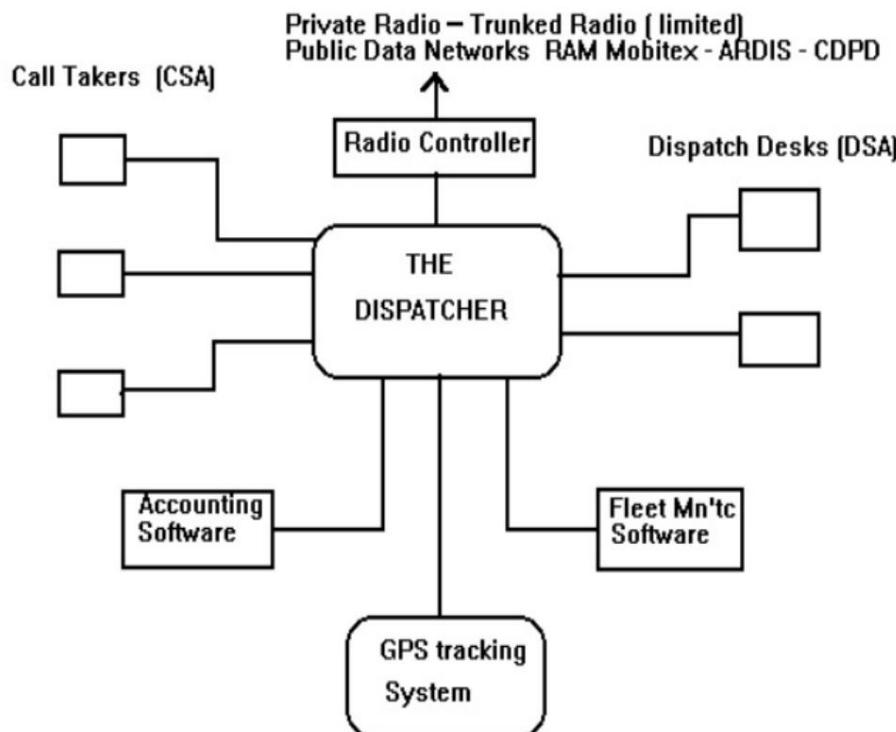
kredi kartı onayı

Mağaza ve süpermarketlerdeki Satış Noktası (POS) terminallerinde, müşterilerin işlemler için kredi kartı kullandığı durumlarda, kart kullanımının doğrulanması için banka merkez bilgisayarı ile POS terminali arasında gerekli olan iletişim hızlı ve güvenli bir şekilde gerçekleşebilmektedir. mobil bilgisayar ünitesi kullanarak hücresel kanallar üzerinden Bu, işlem sürecini hızlandırabilir ve POS terminallerindeki tıkanıklığı giderebilir.

Taksi/Kamyon Gönderimi

Birkaç mobil birim (taksi) ile merkezi olarak kontrol edilen bir sevk memuru fikrini kullanan mobil bilgi işlem, taksilere sevk edilen işin tüm ayrıntılarının verilmesine ve taksilerin nerede olduğunuyla ilgili bilgileri merkezi sevk ofisine iletmesine izin verir.

Bu sistem aynı zamanda güvenli teslimatlarda da son derece kullanışlıdır, örneğin: Securicor. Bu, merkezi bir bilgisayarın tüm mobil güvenli teslimat araçlarından durum bilgilerini takip edebilmesini ve alabilmesini sağlar. Yine, CDPD sisteminin güvenlik ve güvenilirlik özellikleri öne çıkmaktadır.



Şekil 15. Taksi Sevk Ağları

Elektronik Posta/Paging

E-posta göndermek ve okumak için bir mobil birimin kullanılması, herhangi bir iş bireyi için çok yararlı bir varlıktır, çünkü herhangi bir iş arkadaşıyla ve işlerini etkileyebilecek acil gelişmelerle iletişim halinde olmasını sağlar. Mobil bilgi işlem teknolojisini kullanarak internete erişim, bireyin geniş bilgi dizilerine parmak uçlarında sahip olmasını sağlar. Tek bir mobil bilgisayar cihazı kullanarak bireyler arasında daha fazla iletişim yeteneği sağlanır. (4000)

[https://www.brainkart.com/article/Mobile-Computing-Applications_9874/]

3. Yukarıda açıklanan mobil uygulamaların her türünü kendi kelimelerinizle açıklayın.

BİLGİ SAYAR TERİMLERİ SÖZLÜĞÜ

soyutlama. Veri veya işlevin mantıksal özelliklerinin bir bilgisayar programında uygulanmasından ayrılması. Bakınız: kapsülleme, bilgi gizleme, yazılım mühendisliği i.

erişim zamanı. Bir veri çağrısının başlatıldığı an ile verinin tesliminin tamamlandığı an arasındaki zaman aralığı i.

kesinlik.) (1) Doğruluğun veya hatasızlığın niteliksel bir değil erlendirmesi. (2) Hatanın büyüklüğünün nicel bir ölçüsü. Hassasiyetle kontrast. (3) Bir aracın gerçek veya mutlak bir değil ere yaklaşma yeteneğinin ölçüsü. Kesinlik ve yanılığının bir işlevidir.

algoritma. (1) Bir problemin sonlu sayıda adımda çözümü için sonlu, iyi tanımlanmış kurallar dizisi. (2) Belirli bir görevi gerçekleştirmek için herhangi bir işlem dizisi.

analog cihaz. Basınçlar, dirençler, dönüsler, sıcaklıklar ve voltajlar gibi sürekli olarak ölçülen niceliklerle temsil edilen değil işkenlerle çalışan bir cihaz.

Uygulama yazılımı. Bir kullanıcının özel ihtiyaçlarını karşılamak için tasarlanmış yazılım; örneğin, navigasyon, bordro veya süreç kontrolü için yazılım. Destek yazılımı ile kontrast; sistem yazılımı.

mimari tasarım. (1) Bir bilgisayar sisteminin geliştirilmesi için çerçeveye oluşturmak için bir donanım ve yazılım bileşenleri koleksiyonunu ve bunların arayüzlerini tanımlama süreci. (2) (1)'deki işlemin sonucu.

arşiv veritabanı. Veritabanının kurtarılması veya geri yüklenmesinde kullanılmak üzere önemli bir zamanda kaydedilmiş bir veritabanının geçmiş kopyası.

Arşiv dosyası. Daha sonra araştırma veya doğrulama, güvenlik, tarihsel veya yasal amaçlar veya yedekleme için ayrılan bir dosya koleksiyonunun parçası olan dosya.

aritmetik mantık Birimi. Bir bilgisayarın aritmetik ve mantıksal işlemlerini gerçekleştirmekten sorumlu olan CPU içindeki [yüksek hızlı] devreler.

dizi. Tek bir adla ve bir veya daha fazla endeksle tanımlanan n-boyutlu sıralı veri öğeleri kümesi, böylece kümenin her bir öğesi ayrı ayrı adreslenebilir; örneğin bir matris, tablo veya vektör.

montajcı. Assembly dilinde yazılmış programları [kaynak kod dosyaları] makine dili eşdeğeri erlerine [nesne kodu dosyaları] çeviren bir bilgisayar programı. Derleyici, yorumlayıcı ile kontrast.

montaj dili. Belirli bir bilgisayarın komut setine çok yakın bir şekilde karşılık gelen düşük seviyeli bir programlama dili, işlemlerin sembolik olarak adlandırılmasına izin verir ve

adresler ve genellikle program talimatlarının [anımsatıcılar] makine talimatlarına bire bir çevirisisiyle sonuçlanır. Bakınız: düşük seviyeli dil.

asenkron. Düzenli bir zaman ilişkisi olmadan gerç ekleşen, yani zamandan ba ğ ımsız.

BIOS. Temel Giriş Çıkış Sistemi.

TEMEL. Yeni Başlayanlar İçin Kısaltma Çok Amaçlı Sembolik Talimat Kodu, etkileşimli bir ortamda programlamayı öğrenmeyi kolaylaştırmayı amaçlayan üst düzey bir programlama dili.

Temel Giriş Çıkış Sistemi. Bir PC'deki ç evresel aygıtları etkinleştiren bellenim.

Klavye, ekran, disk, paralel ba ğ lantı noktası ve seri ba ğ lantı noktası ve saat ve tarih gibi dahili hizmetler için rutinleri içerir. İşletim sistemindeki aygit sürücülerinden ve uygulama programlarından gelen istekleri kabul eder. Ayrıca, başlangıçta sistemi test eden ve bilgisayarı çalışmaya hazırlayan otomatik başlatma işlevlerini de içerir. İşletim sistemini yükler ve kontrolü ona devreder.

toplu işleme. Etkileşimli işleme olmadan programların seri olarak yürütülmesi. Gerçek zamanlı işleme ile kontrast.

ön yargı. Bir dizi tekrarlanan ölçümdeki ortalama de ğ erin gerçek de ğ ere ne kadar yakın olduğunu bir ölçüsü. Bakınız: doğruluk, kesinlik, kalibrasyon.

ikili. İki tabanlı sayı sistemi. İzin verilen rakamlar "0" ve "1" dir.

biraz. İki basamak teriminin bir daralması. Bit, dijital verilerin temel birimidir. Mantık 1 veya mantık 0 olmak üzere iki durumdan birinde olabilir.

engellemek. (1) Teknik veya mantıksal amaçlarla bir bütün olarak kabul edilen bir dizi kayıt, kelime veya karakter. (2) Bir birim olarak kaydedilen ve birimler arası boşluklarla ayrılan bitişik kayıtlar topluluğu. (3) Programlama dillerinde, bir programın ilgili ifadeleri gruplandırılmasının sınırlandırılmasına, depolama tahsisini belirlemeye, etiketlerin uygulanabilirliğinin tanımlamaya veya programın bölgümlerini başka amaçlar için bölgümlere ayırmaya hizmet eden bir alt bölüm. FORTRAN'da bir blok bir dizi ifade olabilir; COBOL'da fiziksel bir kayıt olabilir.

blok diyagramı. Ana parçaların, hem parçaların temel işlevlerini hem de bunlar arasındaki işlevsel ilişkileri göstermek için uygun şekilde açıklamalı geometrik şekillerle temsil edildiği bir sistem, alet veya bilgisayar diyagramı.

blok uzunluğu. (1) Bir bloktaki kayıt, kelime veya karakter sayısı. (2) Genellikle kayıtlar, sözcükler, bilgisayar sözcükleri veya karakterler gibi birimlerle belirtilen bir bloğun boyutunun ölçüsü.

bot. (1) Belleğin temizleyerek ve işletim sistemini yeniden yükleyerek bir bilgisayar sistemini başlatmak. (2) Bir bilgisayar sisteminin bilinen bir başlangıç durumuna ulaşmasını sağlamak.

önyükleme. Bir bilgisayarda kalıcı olarak bulunan veya kolayca yüklenen ve yürütülmesi, işletim sistemi veya yükleyicisi gibi daha büyük bir programı belleğe getiren kısa bir bilgisayar programı.

tampon. Veri akış oranlarındaki, olayların meydana geldiği zamandaki veya verilerin aktarılmasında veya kullanılmasında yer alan cihazlar veya süreçler tarafından işlenen veri miktarlarındaki farklılıklar telafi etmek için verileri geçici olarak depolamak için kullanılan bir cihaz veya depolama alanı [bellek]. veri.

böcek. Programın istenmeyen veya beklenmeyen bir şekilde çalışmasına neden olan bir program hatası. Bakınız: anormallik, kusur, hata, istisna, kusur.

otobüs. Bir bilgisayar sistemi içindeki farklı donanım aygıtları arasında veri ve kontrol sinyallerinin yol aldığı ortak bir yol.

bayt. Bir birim olarak çalıştırılan, genellikle sekiz olan bitişik bit dizisi.

KAM. bilgisayar destekli üretim. Bir iş parçasının üretilmesi için gerekli olan işleme [sayısal kontrol, süreç kontrolü, robotik, malzeme ihtiyaç planlaması] için makineleri otomatikleştirmek için iş talimatlarını iletmek üzere bilgisayarların kullanımı dahil olmak üzere üretim sistemlerinin ve tekniklerinin otomasyonu.

DAVA. Bilgisayar Destekli Yazılım Mühendisliği. Proje planlama ve tahmin, sistem ve yazılım gereksinimleri analizi, veri yapısının tasarımları, program mimarisinin ve yazılım mühendisliğinin yöntemlerinin ve görevlerinin yerine getirilmesini kolaylaştıran programlar, yani entegre bir araç seti içeren yazılım geliştirmenin desteklenmesi için otomatik bir sistem. algoritma prosedürü, kodlama, test ve bakım.

COTS. yapılandırılabilir, kullanıma hazır yazılım. Uygulama yazılımı, bazen genel amaçlı, eşitlik endüstriler veya kullanıcılar için, kullanıcıların programı kendi bireysel ihtiyaçlarını karşılayacak şekilde değil istirmesine izin verecek şekilde yazılmış.

İŞLEMCİ. Merkezi işlem birimi. Program talimatlarının yorumlanması ve yürütülmesini kontrol eden devreleri içeren bir bilgisayar birimi. CPU tüm bilgisayarı kontrol eder. Giriş-çıkış kanalları aracılığıyla veri alır ve gönderir, verileri ve programları bellekten alır ve bir programın matematiksel ve mantıksal işlevlerini yürütür.

C. Genel amaçlı bir üst düzey programlama dili. Bilgisayar işletim sistemleri yazılımının geliştirilmesinde kullanılmak üzere oluşturulmuştur. Assembly dilinin gücünü üst düzey bir dilin kolaylığı ile birleştirmeye çalışır.

C++. Nesne yönelimli üst düzey bir programlama dili.

takipçi değil iştir Bir programda yapılan tüm değil işiklikleri belgeleyen bir yazılım aracı.

müşteri sunucusu. Bir hizmetin alıcısı ile sağlayıcısı arasındaki ilişkiyi tanımlamak için geniş anlamda kullanılan bir terim. Mikrobilgisayar dünyasında, terim

istemci-sunucu, istemci olarak ön uç uygulamaların başka bir ağ bağ lantılı sistem üzerinde hizmet istekleri yaptığı i ağ bağ lantılı bir sistemi tanımlar.

COBOL. Ortak İ ş Odaklı Dilin kısaltması. İ ş verilerinin işlenmesindeki sorunların çözümünde kullanılması amaçlanan üst düzey bir programlama dili.

kod denetimi. Yazılım tasarım belgeleri ve programlama standartlarına uygunluğu doğ rulamak için bir kişi, ekip veya araç tarafından kaynak kodun bağımsız bir incelemesi.

kod incelemesi. Programcının, program mantığıını analiz eden sorular soran, kodu tarihsel olarak yaygın programlama hatalarının bir kontrol listesine göre analiz eden ve analiz eden bir gruba kaynak kodu, ifadeye göre okuduğu manuel [resmi] test [hata tespiti] teknik i. kodlama standartlarına uygunluk.

kodlama. (1) Yazılım mühendisliği içinde, bir bilgisayar programını bir programlama dilinde ifade etme süreci. (2) Tasarım özelliklerinden (tasarım açıklamaları) gelen mantık ve verilerin bir programlama diline dönüştürülmesi.

karşılaştırıcı. Ortak noktaları veya farklılıklarını belirlemek için iki bilgisayar programını, dosyayı veya veri kümesini karşılaştırın bir yazılım aracı. Tipik karşılaştırma nesneleri, kaynak kodun, nesne kodunun, veri tabanı dosyalarının veya test sonuçlarının benzer sürümleridir.

uyumluluk. Belirli bir arabirimin gereksinimlerini karşılamak için işlevsel bir birimin yeteneği i.

derleme. Probleme yönelik bir dilde veya prosedüre yönelik bir dilde ifade edilen bir programı nesne koduna çevirmek.

derleyici. (1) Üst düzey bir dilde ifade edilen programları makine dilindeki eşdeğ erlerine çeviren bir bilgisayar programı. (2) Derleyici, tamamlanmış kaynak kodu listesini girdi olarak alır ve bilgisayarın programı yürütmek için sahip olması gereken makine kodu talimatlarını verir.

bilgisayar. Bir çalışma sırasında insan müdahalesi olmadan sayısız aritmetik işlem veya mantık işlemi de dahil olmak üzere önemli hesaplamaları gerçekleştirebilen işlevsel bir birim.

Bilgisayar destekli tasarım. Ürün tasarlamak için bilgisayar kullanımı. CAD sistemleri, önerilen ürünlerin kullanımını modellemek ve simüle etmek için CAD yazılımı ve grafik tabletler ve tarayıcılar gibi giriş cihazları kullanan yüksek hızlı iş istasyonları veya kişisel bilgisayarlardır.

bilgisayar dili. İnsanların bilgisayarlarla iletişim kurmasını sağlamak için tasarlanmış bir dil. Bakınız: programlama dili.

bilgisayar sistemi. Bir veya daha fazla bilgisayar ve ilişkili çevreSEL giriş ve çıkış aygıtlarından ve ilgili yazılımdan oluşan işlevsel bir birim.

yapıldırma. (1) Bir bilgisayar sistemi veya bileşeninin, kendisini oluşturan parçalarının sayısı, doğası ve ara bağımlılıkları ile tanımlandığı şekilde düzenlenmesi. (2) Konfigürasyon yönteminde, teknik belgelerde belirtilen veya bir ürününde elde edilen donanım veya yazılımın işlevsel ve fiziksel özellikleri.

kontrol akışı. Programlama dillerinde, bir yürütme dizisinin bir program boyunca alabileceğin tüm olası yolların bir soyutlaması.

kontrolör. Disk veya ekran gibi çevresel aygıtları kontrol eden donanım. Ana bellek ile çevresel aygit arasındaki fiziksel veri transferlerini gerçekleştirir.

çapraz derleyici. Bir bilgisayarda çalışan ancak farklı bir bilgisayar için derleme kodu veya nesne kodu oluşturan bir derleyici.

DOS. disk işletim sistemi.

veri. Gerçeklerin, kavramların veya talimatların iletişim, yorumlama veya insanlar tarafından veya otomatik araçlarla işlenmesi için uygun bir şekilde temsili.

veri analizi. Her birinin program tarafından doğrudan tanımlandığından ve kullanıldığından emin olmak için koddaki veri yapısının ve kullanımının değerlendirilmesi. Genellikle mantık analizi ile birlikte gerçekleştirilir.

veri yolu. Verileri dahili ve harici olarak bir işlem birimine veya bir depolama aygıtına iletmek için kullanılan bir veri yolu.

veri öğesi. (1) Bazı bağlantımlarda bölünemez olarak kabul edilen ve diğer bağlantımlarda veri öğelerinden oluşabilen adlandırılmış bir veri birimi. (2) Bir veritabanında temsil edilen varlıkların her birinin ve niteliklerinin adlandırılmış tanımlayıcısı.

veri akışı analizi. Giriş ve çıkış verilerinin ve biçimlerinin doğrudan tanımlandığından ve veri akışlarının doğrudan emin olmak için bir yazılım V&V görevi.

veri bütünlüğü. Bir veri koleksiyonunun eksiksiz, tutarlı ve doğrudu olma derecesi.

veri seti. İlgili kayıtların bir koleksiyonu. Syn: dosya.

veri yapısı. Belirli veri işleme işlevlerini desteklemek için tasarlanmış veri öğeleri arasında fiziksel veya mantıksal bir ilişki.

veri doğrulama. Verilerin yanlış, eksik veya mantıksız olup olmadığıını belirlemek için kullanılan bir süreç. Süreç, format kontrollerini, eksiksizlik kontrollerini, kontrol anahtarları testlerini, makullük kontrollerini ve limit kontrollerini içerebilir.

veri tabanı. Bir veya daha fazla uygulamaya hizmet etmek için bir şemaya göre düzenlenen, genellikle kontrollü artıklıkla, birbiriyile ilişkili veriler topluluğu. Veriler, veri yapısı veya organizasyonu ile ilgilenmeden farklı programlar tarafından kullanılabilcek şekilde saklanır.

ölü kod. Program çalışması sırasında asla yürütülemeyen program kodu ifadeleri. Bu tür bir kod, zayıf kodlama stilinden kaynaklanabilir veya önceki sürümlerin veya hata ayıklama çabalarının bir eseri olabilir. Ölü kod kafa karıştırıcı olabilir ve hatalı yazılım değil işikliklerinin potansiyel bir kaynağıdır.

hata ayıklama. Bir program hatasının tam niteliğini ve yerini belirleme ve hatayı düzeltme.

tasarım. Bir sistem veya bileşenin mimarisini, bileşenlerini, arayüzlerini ve diğer özelliklerini tanımlama süreci.

tasarım aşaması. Mimari, yazılım bileşenleri, arayüzler ve veriler için tasarımların oluşturulduğu, belgelendiği ve gereksinimleri karşılamak için doğrudan yazılım yaşam döngüsü içinde geçen süre.

tasarım gereksinimi. Bir sistem veya sistem bileşeninin tasarımını belirleyen veya kısıtlayan bir gereksinim.

tasarım standartları. Bir tasarımın özelliklerini veya veri veya program bileşenlerinin tasarım açıklamasını tanımlayan standartlar.

geliştirici. Bilgisayarlı sistemlerin donanımını ve/veya yazılımını tasarlayan ve/veya oluşturan ve/veya belgeler ve/veya yapılandıran kişi veya grup.

geliştirme metodolojisi. Geliştirme aşamalarını tanımlayan ve her aşama için faaliyetleri, ürünleri, doğrulama prosedürlerini ve tamamlama kriterlerini belirleyen yazılım oluşturmaya yönelik sistematik bir yaklaşım.

farklı yazılım sistem analizi. Güvenlikle ilgili entegrasyon ve arayüz hatalarını azaltmak için yazılım gereksinimlerinin ayrı bilgisayar sistemlerine tahsisinin analizi. Birden fazla yazılım sistemi entegre edildiğiinde gerçekleştirilebilir.

dijital. Ayrık integral değil erler şeklinde veri [sinyaller] ile ilgili.

disk işletim sistemi. Bir işletim sistemi programı; örneğin, Digital Research'ten DR-DOS, Microsoft Corp.'tan MS-DOS, IBM'den OS/2, IBM'den PC-DOS, Apple'dan System-7.

sürücü. Bir evresel aygıtı veya dahili işlevi işletim sistemine bağlayan ve tüm aygit işlevlerinin etkinleştirilmesini sağlayan bir program.

dinamik analiz. Program kodunu yürüterek gerçekleştirilen analiz.

düzenleme. Karakter, sayı veya veri ekleyerek, silerek veya taşıyarak girişin içeriğini değiştirmeye.

gömülü yazılım. Daha büyük bir sistemin parçası olan ve o sistemin bazı gereksinimlerini yerine getiren yazılım; örneğin, bir uç akta veya hızlı geçiş sisteminde kullanılan yazılım. Bu tür bir yazılım, kullanıcı ile bir arayüz sağlayamaz. Bakınız: bellenim.

kapsülleme. Bir modül içindeki bir sistem işlevini veya bir dizi veriyi ve bu veriler üzerindeki işlemler izole etmekten ve modül iç in kesin özellikler sağ lamaaktan oluşan bir yazılım geliştirme teknig i.

son kullanıcı. Bilgi alışverişinde veri işleme amacıyla bir bilgi sistemini kullanan bir kişi, cihaz, program veya bilgisayar sistemi.

hata. Hesaplanan, gözlemlenen veya ölçülen dē er veya koşul ile gerçek, belirtilen veya teorik olarak doğ ru dē er veya koşul arasındaki tutarsızlık.

hata tespiti. Veri aktarımlarındaki hataları belirlemek için kullanılan teknikler. Bakınız: kontrol toplamı, döngüsel artıklık kontrolü [CRC], eşlik kontrolü, uzunlamasına artıklık.

yürütmeye izi. Bir bilgisayar programının yürütülmesi sırasında yürütülen komut dizisinin kaydı. Genellikle program yürütürken karşılaşılan kod etiketleri listesi şeklini alır.

FTP. dosya aktarım Protokolü.

arıza. Bir sistem veya bileşenin, belirtilen performans gereksinimleri dahilinde gerekli işlevlerini yerine getirememesi.

başarısızlık analizi. Hatayı düzeltmek, dīg er benzer hataları belirlemek ve düzeltmek ve gelecekte bu tür hataların oluşmasını önlemek için düzeltici eylemi başlatmak için bir program hatasının tam yapısını ve yerini belirlemek.

arıza. Bir bilgisayar programında, programın istenmeyen veya beklenmeyen bir şekilde çalışmasına neden olan yanlış bir adım, işlem veya veri tanımı.

dosya. (1) Bir birim olarak ele alınan bir dizi ilgili kayıt;örneğ in, stok kontrolünde, bir dosya bir dizi faturadan oluşabilir. (2) Belirli bir kayıt türündeki kayıtların bir veritabanındaki tüm oluşumların adlandırılmış bir koleksiyonundan oluşan en büyük depolama yapısı birimi.

dosya bakımı. Veri ekleyerek, dē iştirerek veya silerek bir dosyayı güncel tutma etkinliği i.

dosya aktarım Protokolü. (1) İ kili ve ASCII veri dosyalarını veri kaybı olmadan iletebilen iletişim protokolü. Bakınız: Kermit, Xmodem, Ymodem, Zmodem. (2)

Ağ da oturum aç mak, dizinleri listelemek ve dosyaları kopyalamak için kullanılan TCP/IP protokolü. Ayrıca ASCII ve EBCDIC arasında çeviri yapabilir.

bellenim. Bir donanım aygıtının kombinasyonu;örneğ in, bir IC; ve o cihazda salt okunur yazılım olarak bulunan bilgisayar talimatları ve verileri. Bu tür yazılımlar, işleme sırasında bilgisayar tarafından dē iştirilemez.

FORTTRAN. Yaygın olarak kullanılan ilk üst düzey programlama dili olan FORmula TRANslator'ın kısaltmasıdır. Öncelikle matematik, mühendislik ve bilimdeki teknik problemlerin çözümünde kullanılmak üzere tasarlanmıştır.

fonksiyonel Analiz. Her bir güvenlik açısından kritik yazılım gereksiniminin kapsandığıını ve her bir yazılım ögesine uygun bir kritiklik düzeyi atandığını doğrular.

fonksiyonel konfigürasyon denetimi. Bir konfigürasyon ögesinin geliştirilmesinin tatmin edici bir şekilde tamamlandığını, ögesinin işlevsel veya tahsisli konfigürasyon tanımlamasında belirtilen performans ve işlevsel özelliklere ulaştığını ve operasyonel ve destek belgelerinin eksiksiz ve tatmin edici olduğunu doğrulamak için yürütülen bir denetim.

gigabayt. Yaklaşık bir milyar bayt; tam olarak 230 veya 1.073.741.824 bayt.

grafik. Kenarlar veya yaylar olarak adlandırılan sonlu düğümler ve düğümler arasındaki bağlantıları oluşturan bir diyagram veya başka bir temsil.

grafik yazılım özellikleri. Program yapısını, veri durumlarını, kontrolü, işlem akışını, HIPO'yu ve neden sonuç ilişkilerini gösteren çizelgeler, diyagramlar, grafikler gibi belgeler; ve tasarım bütünlüğünü oluşturmak için gerekli doğruluk, karar, olay, durum geçışı, modül arayüzü, istisna koşulları yanıtları içeren tablolar.

donanım. Programların, prosedürlerin, kuralların ve ilgili belgelerin aksine fiziksel ekipman.

üst düzey dil. Hedef bilgisayar hakkında çok az bilgi gerektiren, birkaç farklı makine diline çevrilebilen, işlemlerin ve adreslerin sembolik olarak adlandırılmasına izin veren, veri yapılarının ve program mantığıının ifadesini kolaylaştmak için tasarlanmış özellikler sağlayan ve genellikle birkaç makine talimatıyla sonuçlanan bir programlama dili. her program ifadesi. Önekler PL/1, COBOL, BASIC, FORTRAN, Ada, Pascal ve "C"dir.

ben/0. giriş çıkış.

ISO. Uluslararası Standardizasyon Örgütü.

uygulama. Bir tasarımını donanım bileşenlerine, yazılım bileşenlerine veya her ikisine çevirme süreci.

uygulama aşaması. Tasarım belgelerinden bir yazılım ürününün oluşturulduğu ve hata ayıklanlığı yazılım yaşam döngüsündeki süre.

uygulama gereksinimi. Bir sistem veya sistem bileşeninin kodlamasını veya yapısını belirleyen veya kısıtlayan bir gereklilik.

artımlı gelişme. Gereksinim tanımı, tasarım, uygulaması ve testinin örtüsü, [sıralı yerine yinelemeli] bir şekilde gerçekleşti ve genel yazılım ürününün aşamalı olarak tamamlanmasıyla sonuçlanan bir yazılım geliştirme tekniği.

Bilgi gizleme. Bir işlevin veya yapının ayrıntılarını programın dış er böülümlerine erişilemez hale getirerek "gizleme" uygulaması.

giriş çıkış. Her mikroişlemci ve her bilgisayar, programları iç in gerekli verileri elde etmek ve veri manipülasyonlarının sonuçlarını iletmek iç in dış dünya ile iletişim kurmanın bir yoluna ihtiyaç duyar. Bu, I/O portları ve cihazları aracılığ ıyla gerç ekleştirilir.

Kurulum. Bilgisayarlı bir sistemin donanım ve yazılımının montajını ve test edilmesini iç eren sistem yaşam döngüsündeki aşama. Kurulum, yeni bir bilgisayar sistemi, yeni yazılım veya donanım kurmayı veya mevcut sistemin başka bir şekilde deň istirilmesini iç erir.

kurulum ve kontrol aşaması. Bir yazılım ürününün işletim ortamına entegre edildiğ i ve gerektiğ i gibi ç alışmasını sağ lamak iç in bu ortamda test edildiğ i yazılım yaşam döngüsü iç inde geçen süre.

talimat. (1) bir bilgisayarın belirli bir işlemi veya işlem dizisini gerç ekleştirmesine neden olan program ifadesi. (2) Bir programlama dilinde, bir işlemi belirten ve varsa işlenenlerini tanımlayan anlamlı bir ifade.

entegre devre (IC). Son derece küçük elektronik anahtarlama devreleriyle kazınmış veya basılmış yarı iletken malzemeden [silikon] küçük gofretler. Sin: ç ip.

arayüz. İ şlevsel özellikler, ortak fiziksel ara bağ lantı özellikler, sinyal özellikler ve uygun olduğ u şekilde dış er özellikler tarafından tanımlanan iki işlevsel birim arasında paylaşılan bir sınır. Konsept, farklı işlevlere sahip iki cihazın bağ lantısının belirtilmesini iç erir.

yorumlamak. Bir bilgisayar programının her bir ifadesini veya yapısını, bir sonrakini tercüme etmeden ve ç alıştırmadan önce tercüme etmek ve yürütmemek.

tercüman. Bir bilgisayar programının her bir ifadesini veya yapısını, bir sonrakini tercüme etmeden ve ç alıştırmadan önce çeviren ve yürütmen bir bilgisayar programı. Yorumlanan bir dilde yazılmış bir program [kaynak kod dosyası] her yürütüldüğ unde, yorumlayıcı bilgisayarda yerleşik olmalıdır.

geçersiz girişler Programın temsil ettiğ i fonksiyonun etki alanı dışında kalan test verileri.

KB. kilobayt. Yaklaşık bin bayt. Bu simbol, bilgisayar belleğ inin veya disk depolama alanının boyutunu belirtmek iç in kullanılır. Bilgisayarlar ikili sayı sistemi kullandığı ından, bir kilobayt tam olarak 210 veya 1024 bayttır.

anahtar unsur. Üretim sürecinin kritik bir kontrol noktasında tek bir adım.

yaşam döngüsü metodolojisi. Planlamak, tasarlama, uygulamak, test etmek için çeşitli yapılandırılmış yöntemlerden herhangi birinin kullanımı. ve bir sistemi konseptinden kullanımının sona ermesine kadar işletmek. Bakınız: şelale modeli.

bağ lantı düzenleyici Modüller arasındaki çapraz referansları çözümleyerek ve muhtemelen öğeleri yeniden konumlandırarak bağ ımsız olarak evrilmiş iki veya daha fazla nesne modülünden veya yük modülünden tek bir yük modülü oluşturan bir bilgisayar programı. Syn: bağ lantı düzenleyici yükleyici. Yürütlmeden önce diğ er [nesne] programları yardımcı [harici] bellekten ana [dahili] belleğ e kopyalayan bir program.

düşük seviyeli dil. Assembly dilinin avantajı, programın optimum hız ve performans için ayarlanmasına izin veren işlemcinin bit düzeyinde kontrolünü sağlamasıdır. Zaman açısından kritik işlemler için, gerekli işlemler için yeterince hızlı çalışan kod üretmek için montaj dili gereklidir. Montaj dilinin dezavantajı, programlamada gereken yüksek düzeyde karmaşıklık ve ayrıntıdır. Bu, kaynak kodunun anlaşılması zorlaştırır, böylece program geliştirme ve bakım sırasında hata verme olasılığını artırır.

Mb. megabit. Yaklaşık bir milyon bit. Tam olarak 1024 K bit, 220 bit veya 1.048.576 bit.

MB. megabayt. Yaklaşık bir milyon bayt. Tam olarak 1024 K Bayt, 220 bayt veya 1.048.576 bayt. Bakınız: kilobayt.

MIPS. saniyede milyon talimat.

makine kodu. Bir bilgisayarın CPU'su tarafından tanınable bir biçimde [ikili kod] ifade edilen bilgisayar talimatları ve tanımları. Programlandığı dilden bağ ımsız olarak tüm kaynak kodları, sonunda makine koduna dönüştürülür. Syn: nesne kodu.

makro talimat. Genellikle programın geri kalıyla aynı dilde ve genellikle derleme veya derleme sırasında önceden tanımlanmış bir dizi kaynak talimatla değil istirilen bir kaynak kodu talimatı.

ana hafıza. Bilgi depolamak için bir dizi elektronik devre türünden birini kullanan hareketsiz bir depolama cihazı.

ana program. Bir bilgisayarın işletim sistemi tarafından çağırılan ve genellikle diğ er yazılım bileşenlerini çağırın bir yazılım bileşeni. Bakınız: rutin, alt program.

bakım. Gereksinimlere uygun performansı sağlamak için ekipmanın ayarlanması, temizlenmesi, değil istirilmesi, elden geçirilmesi gibi faaliyetler. Bir yazılım sisteminin bakımı, yazılım hatalarını düzeltmeyi, yazılımı yeni bir ortama uyarlamayı veya yazılımda iyileştirmeler yapmayı içerir.

ölçüm. Bir yazılım ürününün veya sürecinin belirli bir niteliğe sahip olma derecesinin nicel bir değerlendirmesi.

megahertz. Saniyede bir milyon döngüye eşit bir frekans birimi.

hafıza. İkili verilerin saklanabileceğine ve tutulabileceğine tüm orijinal verilerin alınabileceğine herhangi bir cihaz veya kayıt ortamı.

Menü. Bir dizi seçenekleri listeleyen bir bilgisayar ekranı; örneğin, operatörün bir tanesini seçebileceğin fonksiyonlar. Bazen bir program listesini belirtmek için kullanılır.

metrik, yazılım kalitesi. Yazılımın kalitesini etkileyen belirli bir niteliğe sahip olma derecesinin nicel bir ölçüsü.

mikro kod. Bir bilgisayarın komut setindeki her komut için gerçek ekleştirmesi gereken temel devre işlemlerini tutan kalıcı bellek.

mikroişlemci. Tek bir IC'de bulunan bir CPU. Genellikle bir mikrobilgisayar ile eş anlamlıdır.

modelleme. Algoritmik süreçlerin duyarlı hedefler üzerindeki etkileri için bir problemin boyutlarını araştırmak için varsayılan bir ortamın etkilerini modellemek için kullanılan programların inşası.

modül. (1) Programlama dillerinde, bir programın ayrı olarak derlenebilen bağımsız bir alt bölümü. (2) Bir montajçı, bir derleyici, bir bağlantı düzenleyicisi veya benzer bir rutin veya alt rutin tarafından genellikle bir birim olarak işlenen ayrı bir talimat seti. (3) Diğer bileşenlerle kullanıma uygun, paketlenmiş işlevsel bir donanım birimi. Bakınız: birim.

çoklu işlem. İki veya daha fazla işlemin [programının] aynı anda [aynı anda] ortak bir ana belleğe erişimi olan ayrı CPU'lar tarafından yürütüldüğü bir çalışma modu. Çoklu programlama ile kontrast.

çoklu programlama. İki veya daha fazla programın tek bir CPU tarafından aralıklı olarak yürütüldüğü bir çalışma modu.

çoklu görev. İki veya daha fazla görevin aralıklı olarak yürütüldüğü bir işlem modu.

ağ. (1) Düğümlerin ve birbirine bağlı olan dalların bir düzenlenmesi. (2) Uzaktaki birkaç bilgisayarı telekomünikasyon yoluyla birbirine bağlı olan bir sistem [iletim kanalları ve destekleyici donanım ve yazılım].

OOP. nesne yönelimli programlama. Belirli bir veri yapısını işlemek için gereken tüm bilgileri içeren kendi kendine yeterli modüllerden oluşan programlar yazmak için bir teknoloji.

Modüller, bir sınıfın kodunun veya yöntemlerinin diğer modüllere geçirilebilmesi için sınıf hiyerarşilerinde oluşturulur. Yeni nesne modülleri, mevcut sınıfların özelliklerini devralarak kolayca oluşturulabilir.

nesne. Nesne yönelimli programlamada, verilerin kendi kendine yeten bir modülü [kapsülleme] ve bu verileri [işleyen] programlar [hizmetler].

nesne kodu. Normalde bir bilgisayar tarafından yürütülmeye hazır olan belirli bir çeviri işleminin çıktısı olan makine dilinde ["1"s ve "0"s] ifade edilen bir kod.

nesne yönelimli tasarım. Bir sistem veya bileşenin nesneler ve bu nesneler arasındaki bağ lantıları cinsinden ifade edildiği bir yazılım geliştirme teknigi i.

nesne yönelimli dil. Kullanıcının bir programı nesneler ve bu nesneler arasındaki mesajlar açısından ifade etmesini sağlayan bir programlama dili. Önekler arasında C++, Smalltalk ve LOGO sayılabilir.

nesne programı. Bir derleyici veya derleyicinin çıktısı olan bir bilgisayar programı.

İşletim sistemi. Programların yürütülmesini kontrol eden ve kaynak tahsis, zamanlama, giriş/çıkış kontrolü ve veri yönetimi gibi hizmetler sağlayan yazılım. Genellikle işletim sistemleri ağ ıraklı olarak yazılımdır, ancak kısmi veya tam donanım uygulamaları mümkündür.

Oracle. SQL programlama dilini içeren ilişkisel bir veritabanı programlama sistemi. Oracle Corp.'un tescilli ticari markasıdır.

Orijinal Ekipman Üreticisi. Bilgisayar donanımı üreticisi.

BALO. programlanabilir salt okunur bellek.

paralel. (1) İki veya daha fazla işlemin eşzamanlılığı i ile ilgili. (2) Bir karakterin bitleri veya bir kelimenin karakterleri gibi bir bütünü tek tek parçalarının, çeşitli parçalar için ayrı kolaylıklar kullanılarak eşzamanlı olarak işlenmesiyle ilgili.

(3) Bir karakteri oluşturan bitlerin, genellikle sekiz bit [bir bayt] eşzamanlı iletimini tanımlayan terim.

paralel işleme Bakınız: çoklu işlem, çoklu programlama.

parametre. Yazılım modüllerleri arasında değerleri iletmek için kullanılan bir sabit, değer işken veya ifade.

parite. Bit modellerinin tek sayıda 1 bit [tek eşlik] veya çift olmasına neden olmak için bit modellerine [kelime, bayt, karakter, mesaj] seçici olarak 1 bit eklemekten oluşan veri iletimlerinde bir hata algılama yöntemi. 1 bit sayısı [çift eşlik].

Pascal. Yapılandırılmış programlama uygulamalarını teşvik etmek için tasarlanmış üst düzey bir programlama dili.

yol. Bir bilgisayar programının yürütülmesi sırasında gerçekleştirebilecek bir dizi talimat.

çevre aygıtı. Doğrudan bir bilgisayara bağlı ekipman. Veri girişi için bir çevresel aygit kullanılabilir; örneğin, tuş takımı, barkod okuyucu, döñüştürücü, laboratuvar test ekipmanı; veya çıktı verisi için; örneğin yazıcı, disk sürücüsü, video sistemi, teyp sürücüsü, valf denetleyicisi, motor denetleyicisi. Syn: çevresel ekipman.

piksel. (1) Görüntü işleme ve görüntü tanımada, bir digital görüntünün gri düzeyi atanabilen en küçük öğesi. (2) Bilgisayar grafiklerinde, bağlı ımsız karakteristikler atanabilen bir görüntüleme yüzeyinin en küçük elemanı. Bu terim "resim öğesi" teriminden türetilmiştir.

platform. Bir uygulama programının amaçlandığı şekilde çalışması [gerçekleşmesi] için bulunması ve çalışması gereken donanım ve yazılım. Bir platform, işletim sistemi veya yürütme yazılımı, iletişim yazılımı, mikroişlemci, ağ, giriş/çıkış donanımı, herhangi bir genel yazılım kitaplığı, veritabanı yönetimi, kullanıcı arabirimini yazılımı ve benzerlerini içerir, ancak bunlarla sınırlı değildir.

programı. İşleme için uygun bir talimat dizisi. İşleme, programı yürütmeye hazırlamak için bir derleyici, derleyici, yorumlayıcı veya başka bir çevirmen kullanımını içerebilir. Talimatlar, beyanları ve gerekli beyanları içerebilir.

program tasarım dili. Bir program tasarımını geliştirmek, analiz etmek ve belgelemek için kullanılan, özel yapılara ve bazen doğrulama protokollerine sahip bir belirtim dili.

programlanabilir mantık aygıtı Kullanıcının sitesinde programlanan bir mantık çipi.

programlanabilir salt okunur bellek. Bir PROM programlama cihazı kullanılarak programlanabilen bir çip. Sadece bir kez programlanabilir. Silinemez ve yeniden programlanamaz. Bit konumlarının her biri eriyebilir bir bağlılığıdır.

Programlama dili. Bilgisayar programlarını ifade etmek için kullanılan bir dil.

PROM programcısı. Bir programı [talimatları ve verileri yaz] PROM ve EPROM çiplerine aktarmak için kullanılan elektronik ekipman.

protokol. İşletimin sağlanması sırasında işlevsel birimlerin davranışını belirleyen bir dizi anlamsal ve sözdizimsel kural.

prototipleme. Analiz ve tasarım aşamalarında gerekli işlevlerin belirlenmesini kolaylaştırarak yazılım geliştirme sürecini hızlandırmak için yazılım araçlarını kullanmak. Bu teknikin bir sınırlaması, sistem veya yazılım sorunlarının ve tehlikelerinin tanımlanmasıdır.

sözde kod. Bir yazılım tasarımını ifade etmek için kullanılan programlama dili ve doğral dilin birleşimi. Kullanılırsa, genellikle kaynak kodu yazılmadan önce üretilen son belgedir.

yeterlilik, operasyonel. Proses ekipmanının ve alt sistemlerin, belirlenmiş limitler ve toleranslar dahilinde tutarlı bir şekilde ulaşabileceğine dair güven oluşturmak.

yeterlilik, süreç performansı. Sürecin etkili ve tekrarlanabilir olduğu una dair güven oluşturmak.

VERİ DEPOSU. rasgele erişim belleği. Okunabilir/yazılabilir bellek olarak adlandırılabilircek çipler, içinde saklanan veriler okunabilir veya bu çipler üzerindeki herhangi bir bellek adresine yeni veriler yazılabilir. Rastgele erişim terimi, her bir bellek konumuna [genellikle 8 bit veya 1 bayt] rastgele olarak doğrudan [okunabilir veya yazılabilir] erişilebileceği anlamına gelir.

ROM. sadece hafızayı oku. Verilerin yalnızca CPU tarafından okunabileceğine bir bellek yongası. CPU bu belleğe veri saklamayabilir. ROM'un RAM'e göre avantajı, ROM'un programını korumak için güç gerektirmemesidir. Bu avantaj, her tür ROM yongası için geçerlidir; ROM, PROM, EPROM ve EEPROM.

gerçek zamanlı. Hesaplama sonuçlarının harici süreci kontrol etmek, izlemek veya zamanında yanıt vermek için kullanılabilmesi için, harici bir sürecin meydana geldiği fiili süre boyunca hesaplamanın gerçekleştirildiği bir sistem veya çalışma modu ile ilgili. Toplu ile kontrast. Bakınız: konuşma, etkileşim, kesinti, çevrimiçi.

gerçek zamanlı işleme. Bir aktiviteden veya fizikal bir süreçten veri alan, hesaplamalar yapan ve aktivitenin veya sürecin sonucunu [kontrol etmek] etkilemek için yeterince hızlı bir yanıt döndüren hızlı yanaklı [anında yanıt] çevrimiçi bir sistem; örneğin, bir süreç kontrol uygulaması. Toplu işleme ile kontrast.

kayıt. (1) bir birim olarak ele alınan ilgili veri öğeleri grubu. [Bir veri öğesi (alan) bir kaydın bir bileşenidir, bir kayıt bir dosyanın (veritabanı) bir bileşenidir]. ilişkisel veritabanı. Dosyaları gereklilik gibi birbirine bağlı olan veritabanı düzenleme yöntemi. Dosyalar arasındaki ilişkiler, hesap numaraları ve adlar gibi veriler karşılaştırılarak oluşturulur. İlişkisel bir sistem herhangi iki veya daha fazla dosyayı alabilir ve eşleşen kriterleri karşılayan kaytlardan yeni bir dosya oluşturabilir.

güvenilirlik. Bir sistemin veya bileşenin, belirtilen koşullar altında belirli bir süre boyunca gerekli işlevlerini yerine getirme yeteneği.

gereklilik. (1) Bir kullanıcının bir sorunu çözmek veya bir amaca ulaşmak için ihtiyaç duyduğu bir koşul veya yetenek. (2) Bir sözleşme, standart, şartname veya diğer resmi olarak dayatılan belgeleri yerine getirmek için bir sistem veya sistem bileşeni tarafından karşılanması veya sahip olunması gereken bir koşul veya yetenek. (3) (1) veya (2)'deki gibi bir koşul veya yeteneği in belgelenmiş bir temsili.

gereksinimler aşaması. Bir yazılım ürünü için işlevsel ve performans yetenekleri gibi gereksinimlerin tanımlandığı ve belgelendiği yazılım yaşam döngüsü içinde geçen süre.

sağ lamlık. Bir yazılım sisteminin veya bileşeninin geçersiz girdiler veya stresli çevre koşulları varlığında doğrudan şekilde çalışabilme derecesi. Bakınız: yazılım güvenilirliği.

rutin. Diğer programlar ve alt programlar tarafından çağrılan bir alt program. Not: Bu terim, çeşitli programlama dillerinde farklı şekilde tanımlanmıştır.

SOP'ler. standart çalışma prosedürleri.

SQL. Yapılandırılmış sorgu dili.

Emniyet. Öume, yaralanmaya, meslek hastalıkına veya ekipman veya mülkün zarar görmesine veya kaybolmasına veya evreye zarar verebilecek koşullardan muafiyet.

sunucu. Birden çok kullanıcı tarafından paylaşılan bir ağdaki yüksek hızlı bilgisayar. Tüm kullanıcılar tarafından paylaşılan programları ve verileri tutar.

simülasyon. (1) Bir nesnenin davranışını temsil etmek için yürütülebilir bir modelin kullanılması. Hesaplama donanımını test ederken, dış ortam ve hatta kod bölgelerini simüle edilebilir. (2) Bir dizi kontrollü girdi sağlandığında belirli bir sistem gibi davranışan veya çalışan bir model. Öykünme ile kontrast.

yazılım. Bir sistemin işleyişine ilişkin programlar, prosedürler, kurallar ve ilgili belgeler.

yazılım özelliği. Yazılımın doğal, muhtemelen tesadüfi, özelliği, kalitesi veya özelliği; örneğin işlevsellik, performans, nitelikler, tasarım kısıtlamaları, durum sayısı, hatlar veya dallar.

yazılım tasarımını açıklaması. Analizi, planlamayı, uygulamayı ve karar vermeyi kolaylaştırmak için oluşturulmuş bir yazılım temsili. Yazılım tasarım açıklaması, yazılım tasarım bilgilerini iletmek için bir ortam olarak kullanılır ve sistemin bir planı veya modeli olarak düşünülebilir.

yazılım geliştirme süreci. Kullanıcı ihtiyaçlarının bir yazılım ürününe dönüştürülmesi sürecidir. süreç, kullanıcı ihtiyaçlarının yazılım gereksinimlerine dönüştürülmesini, yazılım gereksinimlerinin tasarımına dönüştürülmesini, tasarımın kod içinde uygulanmasını, kodun test edilmesini ve bazen operasyonel faaliyetler için yazılımın yüklenmesini ve kontrol edilmesini içerir.

yazılım belgeleri. Tasarım veya ayrıntıları tanımlayan veya belirten, yetenekleri açıklayan veya çalışma talimatları sağlayan, insan tarafından okunabilir biçimde bilgisayar listeleri ve çıktıları dahil üzere teknik veriler veya bilgiler.

yazılım öğesi. Yazılım geliştirme veya bakım sırasında üretilen veya elde edilen teslim edilebilir veya süreç içi bir belge.

yazılım Mühendisliği i. Yazılımın geliştirilmesi, işletilmesi ve bakımına sistematik, disiplinli, ölçülebilir bir yaklaşımın uygulanması; yani, mühendislik in yazılıma uygulanması.

yazılım öğesi. Kaynak kodu, nesne kodu, iş kontrol kodu, kontrol verileri veya bu öğelerin bir koleksiyonu. Yazılım öğesiyle kontrast.

yazılım yaşam döngüsü. Bir yazılım ürününün tasarlanmasıyla başlayan ve ürünün artık kullanıma hazır olmadığı zaman sona eren zaman dilimi. Yazılım yaşam döngüsü tipik olarak gereksinimler, tasarım, programlama, test etme, kurulum ve çalıştırma ve bakım gibi faaliyetleri gösteren aşamalara bölünür.

yazılım güvenilirliği i. (1) Yazılımın belirli koşullar altında belirli bir süre boyunca bir sistemde arızaya neden olmama olasılığı i. Olasılık, yazılımdaki sisteme girişlerin ve sistemin kullanımının bir fonksiyonudur. Sisteme girişler, varsa mevcut arızalarla karşılaşıp karşılaşmadığıını belirler. (2) Bir programın, belirtilen koşullar altında belirli bir süre boyunca gerekli işlevlerini doğrudu ve tekrarlanabilir bir şekilde yerine getirme yeteneği i.

kaynak kodu. (1) Bir montajcıya, derleyiciye veya başka bir evirmeye giriş için uygun bir biçimde ifade edilen bilgisayar talimatları ve veri tanımları. (2) Bir bilgisayarın bir görevi gerçekleştirmesine neden olan talimatlar [program] listesinin insan tarafından okunabilir versiyonu.

kaynak programı. Bir bilgisayar tarafından yürütülebilmesi için derlenmesi, birleştirilmesi veya başka bir şekilde çevrilmesi gereken bir bilgisayar programı. Bakınız: kaynak kodu.

Şartname. Bir sistemin veya bileşenin gerekliliklerini, tasarımını, davranışını veya diğer özelliklerini ve genellikle bu hükümlerin karşılanması karşılanmadığıını belirleme prosedürlerini eksiksiz, kesin, doğrulanabilir bir şekilde belirten bir belge.

şartname, ürün. Yazılımın yerleşik sürümünü açıklayan bir belge.

şartname, gereksinimler. Bir sistem veya sistem bileşeninin gereksinimlerini belgeleyen bir belirtim. Tipik olarak işlevsel gereksinimleri, performans gereksinimleri, arayüz gereksinimleri, tasarım gereksinimleri [öznitelikler ve kısıtlamalar], geliştirme [kodlama] standartları vb. içerir.

spiral modeli. Tipik olarak gereksinim analizi, ön ve ayrıntılı tasarım, kodlama, entegrasyon ve test gibi kurucu faaliyetlerin, yazılım tamamlanana kadar yinelemeli olarak gerçekleştirildiği bir yazılım geliştirme süreci modeli.

standart çalışma prosedürleri. Üretim ve süreçlerin kontrolünü sağlamak için gerekli olan yazılı prosedürler [normal ve tanımlanmış koşullarda atılması gereken adımları tarif eden ve tanımlayan].

depolama aygıtı. Verilerin veya programların yerleştirilebileceği i, saklanabileceğ i ve alınabileceğ i bir birim.

yapilandırılmış programlama Yapılandırılmış tasarım içeren ve yapılandırılmış programların geliştirilmesiyle sonuçlanan herhangi bir yazılım geliştirme tekniği i.

Yapilandırılmış sorgu dili. İ lişkisel bir veritabanındaki verileri sorgulamak ve işlemek için kullanılan bir dil. Başlangıçta IBM ana bilgisayarları için geliştirilmiş olup, mini ve mikro bilgisayar veritabanı uygulamaları için oluşturulmuş birçok uygulama bulunmaktadır. SQL komutları, bir veri tabanı ile etkileşimli olarak çalışmak için kullanılabilir veya bir veri tabanı ile arayüz oluşturmak için bir programlama dili ile gömülebilir.

alt program. Bir bilgisayar programının ayrı olarak derlenebilir, yürütülebilir bir bileşeni. Not: Bu terim, çeşitli programlama dillerinde farklı şekilde tanımlanmıştır.

alt program Denetimi, onu çağırın programa veya alt programa döndüren bir rutin. Not: Bu terim, çeşitli programlama dillerinde farklı şekilde tanımlanmıştır.

destek yazılımı. Diğer yazılımların geliştirilmesine ve bakımına yardımcı olan yazılımlar; örneğin derleyiciler, yükleyiciler ve diğer yardımcı programlar.

sözdizimi. Bir dildeki sembollerin kelimeler, deyimler, ifadeler ve diğer izin verilenleri oluşturmak için nasıl birleştirileceğiini tanımlayan yapısal veya dilbilgisi kuralları. yapılırlar.

sistem Analizi. Sistemin işlevlerini ve bunların birbirleriyle ve diğer herhangi bir sistemle ilişkisini belirlemek için gerçek veya planlı bir sistemin sistematik olarak incelenmesi.

Sistem tasarımı. Belirtilen gereksinimleri karşılamak için bir sistem için donanım ve yazılım mimarisini, bileşenleri, modülleri, arayızları ve verileri tanımlama süreci.

sistem yaşam döngüsü. Bir sistemin konseptinden kullanımının sona ermesine kadar geçirdiği gelişimsel değişimlerin seyri; örneğin, bir sistemin analizi, edinimi, tasarıımı, geliştirmesi, testi, entegrasyonu, işletimi, bakımı ve modifikasyonu ile ilgili aşamalar ve faaliyetler. Bakınız: yazılım yaşam döngüsü.

sistem yazılımı. (1) Uygulama yazılımının çalıştırılmasını destekleyen uygulamadan bağımsız yazılım. (2) Bir bilgisayar sisteminin ve ilgili programların işletilmesini ve bakımını kolaylaştırmak için tasarlanmış yazılım; örneğin, işletim sistemleri, montajcılar, yardımcı programlar.

terminal. Genellikle bir CRT ekran ve klavye ile donatılmış, bir iletişim kanalı aracılığıyla bir bilgisayara bilgi göndermek ve bilgisayardan bilgi almak için kullanılan bir cihaz.

Öçek. Belirli koşullar altında bir sistem veya bileşenin yürütüldüğü, sonuçların gözlemlendiği veya kaydedildiği ve sistem veya bileşenin bazı yönleriyle ilgili bir değil erlendirmenin yapıldığı bir faaliyet.

test aşaması. Bir yazılım ürününün bileşenlerinin dē erlendirildī i ve entegre edildī i ve gereksinimlerin karşılanıp karşılanmadı̄ inı belirlemek iç in yazılım ürününün dē erlendirildī i yazılım yaşam döngüsündeki süre.

test prosedürü. Tanımlanan her test iç in sonuçların kurulumu, çalıştırılması ve dē erlendirilmesi iç in ayrıntılı talimatlar sunan bir test planından geliştirilen resmi bir belge.

test yapmak. (1) Belirli koşullar altında bir sistemi veya bileşeni çalışma, sonuçları gözleme veya kaydetme ve sistem veya bileşenin bazı yönleri hakkında bir dē erlendirme yapma süreci. (2) Mevcut ve gerekli koşullar, yanı hatalar arasındaki farkları tespit etmek ve yazılım ȫg elerin özelliklerini dē erlendirmek iç in bir yazılım ȫg esini analiz etme süreci.

test, uyumluluk. İ ki veya daha fazla sistemin bilgi alışverī yapabilme yetenē inı belirleme süreci. Geliştirilen yazılımın halihazırda çalışan bir programın yerini aldığı bir durumda, yeni yazılım ile dīg er programlar veya sistemler arasındaki olası karşılaştırılabilirlik sorunlarını dē erlendirmek iç in bir araştırma yapılmalıdır.

test, birim. (1) Bir modülün tipografik, sözdizimsel ve mantıksal hatalar, tasarımının doğ ru uygulanması ve gereksinimlerinin karşılanması iç in test edilmesi. (2) Bir yazılım ȫg esi iç in tasarımın uygulanmasını doğ rulamak iç in yapılan testler; örnē in, bir birim veya modül; veya bir yazılım ȫg eleri koleksiyonu.

zaman paylaşımı. İ ki veya daha fazla kullanıcının, programlarının yürütülmesini serpiştirek aynı bilgisayar sistemi üzerinde aynı anda bilgisayar programlarını yürütmesine izin veren bir çalışma modu. Zaman dilimleme, öncelīg e dayalı kesintiler veya dīg er zamanlama yöntemleriyle uygulanabilir.

yukarıdan aşağı ıya tasarım. En yüksek soyutlama seviyesiyle başlayan ve giderek daha düşük seviyelere doğ ru ilerleyen tasarım metodolojisi ile ilgili.

işlem. (1) Bir dosyanın güncellenmesi gibi bir işleme eylemi iç in açık veya dolaylı olarak çağ rıda bulunan bir komut, mesaj veya girdi kaydı. (2) Son kullanıcı ile etkileşimli bir sistem arasındaki alışveriş. (3) Bir veritabanı yönetim sisteminde, bir depolama yapısının bir veya daha fazla veri ȫg esinin alınması, güncellenmesi, dē iştirilmesi veya silinmesi gibi belirli bir amacı gerçekleştiren bir işleme faaliyeti birimi.

açık. (1) İ ki veya daha fazla olası anlamı olmayan. (2) Farklı yorumlara açık dē ildir. (3) Belirsiz dē il, belirsiz dē il. (4) Açık, kesin, kesin.

birim. (1) Bir bilgisayar yazılım ȫg esinin tasarımında belirtilen, ayrı olarak test edilebilir bir ȫg e. (2) Bir bilgisayar programının mantıksal olarak ayrılabilir bir parçası. Syn: modül.

UNIX. Araştırma ve geliştirmeyi programlamak iç in uygun bir ortam yaratmak üzere Bell Laboratuvarlarında geliştirilmiş çok görevli, çok kullanıcılı (zaman paylaşımı) bir işletim sistemi.

yardımcı program. Bir bilgisayarın işlemlerini genel olarak destekleyen bir bilgisayar programı; örneğ in bir teşhis programı, bir izleme programı, bir sıralama programı.

yardımcı yazılım. Diğ er uygulama yazılımları, işletim sistemi veya sistem kullanıcıları tarafından gereklili görülen bazı genel destek işlevlerini gerçekleştirmek üzere tasarlanmış bilgisayar programları veya rutinleri. Elektronik ortamı biçimlendirme, dosyaların kopyalarını oluşturma veya dosyaları silme gibi genel işlevleri yerine getirirler.

VV&T. doğ rulama, doğ rulama ve test etme.

geçerli giriş Program tarafından temsil edilen fonksiyonun etki alanı içinde yer alan test verileri.

doğ rulama. (1) Belirli bir sürecin, önceden belirlenmiş özelliklerini ve kalite niteliklerini karşılayan bir ürünü tutarlı bir şekilde üreteceğ ine dair yüksek derecede güvence sağ layan belgelenmiş kanıtların oluşturulması.

doğ rulama, yazılım. Bir geliştirme projesinden üretilen nihai programın veya yazılımın kullanıcı ihtiyaç ve gereksinimlerine göre doğ ruluğ unun belirlenmesi. Doğ rulama genellikle yazılım geliştirme yaşam döngüsünün her aşamasının doğ rulanmasıyla gerçekleştirilir.

doğ rulama, yazılım. Genel olarak, geliştirme yaşam döngüsünün her aşamasında ve her aşaması arasında yazılımın tutarlılığıının, eksiksizliğinin ve doğ ruluğ unun gösterilmesi. Bakınız: doğ rulama, yazılım.

virüs. Diğ er programları, kendisinin bir kopyasını içerecek şekilde gizlice değil istiren ve ana bilgisayar programı yürütüldüğünde yürütülen bir program. Bir virus programının yürütülmesi, yıkıcı olabilecek istenmeyen veya istenmeyen işlevler gerçekleştirerek bir bilgisayar sistemini tehlikeye atar.

Bİ Tİ K. geniş alan ağ i.

bekçi zamanlayıcısı. Olası bir arızayı tespit etmek için kullanılan bir tür aralıklı zamanlayıcı.

ÇÖZÜM

"Yazılım Geliştiricileri için Profesyonel İngilizce" kılavuzu, BT uzmanlık alanında belirli amaçlar için İngilizce öğrenmek ve öğrenmek için anahtar sözcüksel ve dilbilgisel materyal sağlar.

Öğretmenin "Profesyonel Yazılım Geliştirme İngilizcesi" ile çalışması ve öğrencilerin bilgilerinin degerlendirilmesi, Ufa Devlet Havacılık Teknik Üniversitesi son ve yüksek lisans öğrencilerinin gerçek ilgi, psikolojik ihtiyaç ve yeteneklerine uygunluğununu göstermiştir. Önerilen ders kitabı, öğrencilerin ve lisansüstü öğrencilerin yabancı dil profesyonel iletişiminin öğrenme sürecini optimize etmenin bir aracı olarak kabul edilmektedir. Devlet Eğitim Standardına ve teknik üniversiteler için yabancı dil programının gerekliliklerine uygun hale getirilmiştir.

Sunumlar ve tartışmalar için önerilen çeşitli görevler, diyaloglar, rol oynamalar ve konular, farklı türdeki konuşma etkinliklerinde dil ve sosyal ve kültürel becerileri geliştirmek için etkili araçlar olduğunu kanıtladı. Gördüğümüz gibi bu ders kitabı, sınıfta eğitimin sürecini organize eden ve bağımsız ders dışı çalışmalar için görevler seçeneğin bir öğretmenin rehberliği içinde çalışmak üzere tasarlanmıştır.

'Yazılım Geliştiricileri için Profesyonel İngilizce'nin öğrencilerin dil yeterliliklerinde ustalaşmada ilginç ve faydalıdır.

REFERANSLAR VE Bİ LGİ KAYNAKLARI

1. Yazılım Mühendisliği Nedir? [Elektronik kaynak]: – URL: <https://www.castsoftware.com/glossary/what-is-software-engineering> tanım-türleri-of-basics-introduction (20.02.22'ye erişildi)
2. Yazılım. [Elektronik kaynak]: URL: <https://searchapparchitecture.techtarget.com/definition/software> (tarih temyziz 25.02.22)
3. İ şletim sistemi, İ şlevleri ve Özellikleri. [Elektronik kaynak]: – URL: [https://medium.com/computing-technology-with-it/temel-bilgiler/isletim-sistemi-islevleri-ve-ozellikleri-c0946e4215c6](https://medium.com/computing-technology-with-it/temel-bilgiler-isletim-sistemi-islevleri-ve-ozellikleri-c0946e4215c6) (28/02/22'ye erişildi)
4. İ şletim sistemi türleri. [Elektronik kaynak]: – URL: <https://searchapparchitecture.techtarget.com/definition/software> URL: (28.02.22'ye erişildi)
5. İ şletim URL'si Türleri: sistemler. [Elektronik kaynak]: – URL: https://www.tutorialspoint.com/operating_system/os_types.htm (1.03.22'ye erişildi)
6. Yazılımı [Elektronik kaynağı]: türleri. URL: <https://www.geeksforgeeks.org/software-and-its-types.htm> (3/3/22'ye erişildi)
7. Kodlama ve Programlama: En Büyük Farklar. [Elektronik kaynak]: – URL: <https://www.lighthouselabs.ca/en/blog/coding-vs-programming> (03.03.22'ye erişildi)
8. Kuantum bilgisayarlar için ilk sezgisel programlama dili. [Elektronik kaynak]: URL: <https://www.sciencedaily.com/references/2020/06/200615115820.htm> (erişim 28.02.22)
9. Beyin bilgisayar programcılığı için nasıl programlanır? [Elektronik kaynak]: URL: <https://www.sciencedaily.com/references/2020/02/200228094729.htm> (8.03.22'ye erişildi)
10. Ian Sommerville. Yazılım Mühendisliği i. Onuncu Baskı. Pearson. 2016. – PP.200-210.
11. Yazılım Mühendisliği Tartışması. [Elektronik kaynak]: – URL: <https://www.goodreads.com/topic/show/18012721-how-to-become-an-uzman-software-engineer-and-get-any-job-you-want> (tarih temyziz 7.04.22)
12. Programlama Dilleri. [Elektronik kaynak]: URL: https://en.wikipedia.org/wiki/Programming_language (Erişildi 8.03.22)
13. Programlamaya giriş. [Elektronik kaynak]: – URL: <https://www.bbc.co.uk/bitesize/guides/zts8d2p/test> (Erişim tarihi: 03/14/22)
14. Tanım: nesne yönelimli programlama. [Elektronik kaynak]: –

URL: <https://www.computerlanguage.com/results.php?definition=nesne>
yö nelimli+programlama (Erişim tarihi: 24/03/22)

15.Bilgisayar Programlama Dili. [Elektronik kaynak]: – <https://www.britannica.com/technology/computer-programming>
URL: programming
dil/ (24/03/22'ye erişildi)

16.Yeni Programcı için TypeScript. [Elektronik kaynak]: – URL: <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html> (20.03.22'ye erişildi)

17.Görsel Temel. [Elektronik kaynak]: <https://www.britannica.com/computer-programming> URL: dil/Visual-Basic.

18.Veri Yapıları ög retici. [Elektronik kaynak]:
URL: <https://www.javatpoint.com/data-structure-tutorial> (erişim tarihi 25.03.22)

19.C Sharp (programlama dili). [Elektronik kaynak]: - URL: [https://en.wikipedia.org/wiki/C_Sharp_\(programming_language\)](https://en.wikipedia.org/wiki/C_Sharp_(programming_language)) (1.04.22'ye erişildi)

20. Web geliştirme. [Elektronik kaynak]: URL: https://en.wikipedia.org/wiki/Web_development#:~:text=Web%20development,-%20Wikipedia%2C%20the'den (1.04.22'ye erişildi)

21.Web geliştirme nedir? [Elektronik kaynak]: <https://careerfoundry.com/en/blog/web-development/> URL: (04/11/22'ye erişildi)

22.Web Sayfasının Anatomisi: 14 Temel Öğe. [Elektronik kaynak]: - URL: <https://blog.tubikstudio.com/anatomy-of-web-page/> (1.04.22'ye erişildi)

23.Uygulama Geliştirme Nedir? - 3 Ana Uygulama Türü
Geliştirme Metodolojileri. [Elektronik kaynak]:
URL: <https://kissflow.com/low-code/rad/types-of-application-development> metodolojileri (3.04.22'ye erişildi) 24. İngilizce. Bilgi teknolojisi = Rusça için

Bilgi Teknolojisi: teknik ve ög retim ög recileri için bir ders kitabı
mühendislik ve ekonomik uzmanlıklar / I. Yu. Vanik, O. A. Lapko, N. V. Suruntovich. - Minsk: BNTU, 2016. - 157 s.

25.Dil [Elektronik kaynak]: <https://www.techtarget.com/searchenterpriseai/definition/Technology/> (20.03.22'ye erişildi)

26.Oyun motoru. [Elektronik kaynak]: https://en.wikipedia.org/wiki/Game_engine (23.02.22'ye erişildi) URL:

27.Yapay zeka. [Elektronik kaynak]: <https://www.techtarget.com/searchenterpriseai/definition/AI-Yapay-Zeka> (Erişim tarihi 23/03/22)

- 28.Dil Türleri. Eğitim Odaklı Diller. [Elektronik kaynak]: – <https://www.britannica.com/>
URL: technology/computer-programming
dil (23/04/22'ye erişildi)
29. OOAD (Nesneye yönelik analiz ve tasarım) nedir? [Elektronik kaynak]: – URL: [https://www.brainkart.com/article/What-is-OOAD\(Nesneye yönelik-analiz-ve-tasarım\)_9969/](https://www.brainkart.com/article/What-is-OOAD(Nesneye yönelik-analiz-ve-tasarım)_9969/)(Erişim tarihi: 23/04/22)
30. Oluşturma (bilgisayar grafikleri). [Elektronik kaynak]: – <https://en.wikipedia.org/wiki/>
URL: Rendering_(computer_graphics)
(5.04.22'ye erişildi)
- 31.Dört Yapay Zeka Türünü Anlamak. [Elektronik kaynak]: – URL: <https://www.govtech.com/computing/understanding-the-four-type-of-artificial-intelligence.html> (Erişim tarihi: 04/16/22)
- 32.Linux Sistemi - Temel Kavramlar. [Elektronik kaynak]: – URL: https://www.brainkart.com/article/Linux-System---Basic-Concepts_9864/ (07/04/22'ye erişildi)
- 33.Müşterilerle İletişim Kurmak İçin Geliştirici Kılavuzu. [Elektronik kaynak]: – URL: <https://inchoo.net/life-at-inchoo/developers-guidecommunication-clients/> (30/03/22'ye erişildi)
- 34.Nesneye Yönelik Programlama. [Elektronik kaynak]: https://www.brainkart.com/article/Object-Oriented_Programming_10384/ (Erişim tarihi: 03/04/22)
- 35.Yazılım kalitesinde sürecin rolü. [Elektronik kaynak]: - URL: https://www.brainkart.com/article/Role-of-process-in-software-quality_9136/ (1.04.22'ye erişildi)
- 36 Mobil Bilgi İşlem Uygulamaları. [Elektronik kaynak]: - URL: https://www.brainkart.com/article/Mobile-Computing_Applications_9874/ (10.04.22'ye erişildi)
37. İngilizce-Rusça ve Rusça-İngilizce sözlük. [Elektronik kaynak]: – URL: <https://www.multitran.com/> 38.Bilgisayar Sözlüğü. [Elektronik kaynak]: URL: https://www.tutorialspoint.com/computer_glossary.htm