++++

**Parallel dasturlashda MPI (Message Passing Interface) qanday ishlaydi?**

====

#Xodimlar o'zaro axborot almashishadi.

====

Klasslar orqali murojaat qiladi

====

Bitta dastur boshqa dastur bilan xabar almashadi

====

Dasturlar qanday ishlashi haqida xodimlarni xabardor qiladi

++++

**Paralel dasturlashda "Race condition" nima?**

====

#Xodimlar parallel ishlayotganda kodni bir-biriga kiritish xavfsizligi.

====

Bitta xodimning boshqa xodimni kechiktirishi

====

Bitta dasturning murojaat qiladigan saqlangan joy

====

Bitta xodim boshqa xodimni uchratadi

++++

**Paralel dasturlashda "Fork-Join" modeli qanday ishlaydi?**

====

#Dastur o'zi parallel ishlayotgan qismlarga bo'linadi, keyin qaytadan birlashtiriladi

====

Bitta dastur boshqa dasturni boshlatadi.

====

Bitta dastur boshqa dastur bilan murojaat qiladi.

====

Bitta xodim boshqa xodimlarning ishlashini boshlaydi.

++++

**SIMD (Single Instruction, Multiple Data) nima uchun ishlatiladi?**

====

#Bitta instruksiya barcha xodimlar tomonidan ishlatiladi.

====

Bitta xodim boshqa xodimlarning ishlashini boshlaydi

====

Xodimlar bir-biriga murojaat qilishadi

====

Bitta dasturning bir nechta o'zgaruvchilarini ishlatish

++++

**Thread pool nima?**

====

#ish tizimini boshqaruvchi, bitta xodimlar pooli

====

Thread'lar to'plami

====

Bitta dastur parallel ishlayotgan bir nechta dasturlardan iborat

====

Bitta xodimlar tomonidan yaratilgan maxsus dastur

++++

**Parallel dasturlashda "Load balancing" nima uchun kerak?**

====

#vazifalarni xodimlar orasida teng ravishda bo'lib taqsimlash

====

Bitta xodimning barcha ishlarni bajarishi

====

Xodimlar ishlarini barchasi bir vaqtning o'zida bajarsin

====

Ishlarni barcha xodimlar o'zi boshqarishi

++++

**Paralel dasturlashda "Critical section" nima?**

====

#ish tizimining muhim qismi, faqat bir xodim bajarishi kerak

====

Barcha xodimlar bitta ishni bajaradi

====

Xodimlar o'zaro almashishadi

====

Dasturda boshqa dasturlarni chaqirish

++++

**Parallel dasturlashda "Thread contention" nima uchun muhim?**

====

#Bitta xodim boshqa xodimni bajarishi

====

xodimlar o'zaro almashishadi

====

barcha xodimlar bir vaqtning o'zida ishlashadi

====

xodimlar ishlarini boshqarishadi

++++

**Paralel dasturlashda "Task parallelism" qanday ishlaydi?**

====

#ishlar parallel dasturlash uchun tayyorlanadi

====

Bitta xodim boshqa xodimni bajarishi

====

Dasturda bir-biriga bog'liq vazifalar ishlatiladi

====

Xodimlar o'zaro almashishadi

++++

**C++ dasturlash tilida paralel dasturlash uchun qanday kutubxona ishlatiladi?**

====

#OpenMP.

====

Stdlib

====

vector

====

Iostream

++++

**Java dasturlash tilida paralel dasturlash uchun qanday vosita ishlatiladi?**

====

#Multithreading

====

Singlethreading;

====

Parallelthreading;

====

Jointthreading;

++++

**Paralel dasturlashda "Barrier synchronization" qanday ishlaydi?**

====

#Xodimlar bitta qatorda to'planadi,

====

Bitta xodim o'zi boshqaradi

====

Xodimlar bitta kod qismida to'planadi

====

Barcha xodimlar bir vaqtning o'zida ishlashadi

++++

**Paralel dasturlashda "Race condition" qanday o'zgaruvchilarda yuzaga keladi?**

====

#Global o'zgaruvchilar;

====

Static o'zgaruvchilar

====

Xavfsiz o'zgaruvchilar

====

Local o'zgaruvchilar

++++

**C++ dasturlash tilida qanday operator paralel dasturlash uchun yaratilgan?**

====

# //

====

\*\*

====

<<

====

\*\*

++++

**Paralel dasturlashda "Data parallelism" qanday ishlaydi?**

====

#Barcha xodimlar o'zaro bog'liq ishlashadi.

====

Bitta xodim boshqa xodimni bajarishi

====

Xodimlar o'zaro axborot almashishadi

====

Parallel dasturlash uchun yaratilgan kod

++++

**"Load balancing" nima uchun muhim paralel dasturlash prinsipi?**

====

#vazifalarni xodimlar orasida teng ravishda bo'lib taqsimlash

====

Xodimlar barcha ishlarni bajaradi

====

Bitta xodimning barcha vazifalarini bajarish

====

Ishlar bir-biriga bog'liq bo'lishi

++++

**Parallel dasturlashda "Task parallelism" qanday o'zgaruvchilarni ishlatadi?**

====

#Local o'zgaruvchilar:

====

Xavfsiz o'zgaruvchilar;

====

Static o'zgaruvchilar;

====

Global o'zgaruvchilar;

++++

**Paralel dasturlashda "Pipeline parallelism" nima?**

====

#Ishlar bitta qatordan boshqa qatorga o'tadi.

====

Bitta xodim boshqa xodimni bajarishi

====

Ishlar bitta qatorda to'planadi

====

Xodimlar o'zaro almashishadi

++++

**Paralel dasturlashda "Task parallelism" va "Data parallelism" qanday farq qiladi?**

====

#"Task parallelism" vazifalarni to'plamish, "Data parallelism" esa ishlovchi ma'lumotlar

====

"Task parallelism" bitta xodim, "Data parallelism" esa ko'p xodimlar.

====

"Task parallelism" bitta kod qismi, "Data parallelism" esa bitta vazifa.

====

"Task parallelism" dasturni boshqa dastur bilan bog'laydi, "Data parallelism" esa o'zaro xodimlar.

++++

**Parallel dasturlashda "Thread contention"ni kamaytirish uchun qanday usullar mavjud?**

====

#Barcha javoblar to'g'ri;

====

Mutex ishlatish

====

Semaphore ishlatish

====

Xavfsiz o'zgaruvchilar ishlatish

++++

**Obyektga yo‘naltirilgan dasturiy ta’minot (OOP) nima uchun ishlatiladi?**

====

#Ish bajarish jarayonlarini modellash

====

Korxona tizimini rivojlantirish

====

Ma'lumotlarni yig'ish

====

Ma'lumotlar tahlili

++++

**Obyektga yo‘naltirilgan dasturiy ta’minotning xususiyatlari?**

====

#Barchasi to'g'ri.

====

Polimorfizm

====

Abstraksiya

====

Inkapsulyatsiya

++++

**C++ dasturlash tilida nima o'zgaruvchilarga yo'naltirilgan dasturiy ta’minot yaratishni ta'minlaydi?**

====

#Klass

====

Funksiya.

====

Massiv.

====

Struktura.

++++

**Obyektga yo‘naltirilgan dasturiy ta’minotda nima asosiy prinsipga asoslangan?**

====

#Inkapsulyatsiya

====

Abstraksiya.

====

Polimorfizm.

====

Interfeys.

++++

**Obyektga yo‘naltirilgan dasturiy ta’minotda polimorfizm nima?**

====

#Bitta amalning bir nechta shakli

====

Obyektlarning bir-biriga bog'liqligi.

====

Ma'lumotlarning yig'indisi.

====

Obyektlarning ichki xususiyatlarini yashirish.

26…………………………………………………………………………

**Obyektga yo‘naltirilgan dasturiy ta’minotda interfeys nima?**

====

#Funksiyalar to'plami

====

Obyektlarning bir-biriga bog'liqligi.

====

Ma'lumotlarning yig'indisi.

====

Obyektlarning ichki xususiyatlarini yashirish.

++++

**C++ dasturlash tili qanday obyektlarni yaratish uchun ishlatiladi?**

====

#Klass.

====

Funksiya

====

Massiv

====

Struktura

++++

**C++ dasturlash tili qaysi paradigmalarga mos keladi?**

====

#Imperativ

====

Deklarativ

====

Funksional

====

Barchasi to'g'ri

++++

**C++ dasturlash tilida interfeys yaratish uchun nima ishlatiladi?**

====

#Klass

====

Funksiya

====

Massiv

====

Struktura

++++

**C++ dasturlash tili qanday protsedurani (procedure) ifodalash uchun ishlatiladi?**

====

#Funksiya.

====

Klass

====

Massiv

====

Struktura

++++

**C++ dasturlash tili qanday murakkab dasturlarni yaratish imkoniyatiga egami?**

====

#Ha, C++ murakkab dasturlarni yaratish uchun juda qulaydir.

====

Yo'q, C++ dasturlash tilini murakkab dasturlarni yaratishda ishlatib bo'lmaydi

====

Faqatgina operatsion tizimlar uchun

====

Aksariyat dasturlar C++ tilida murakkab bo'lmaydi

++++

**Obyektga yo‘naltirilgan dasturiy ta’minotda nima obyekt hisoblanadi?**

====

#Real obyekt

====

Obyektiv funksiya

====

Blok

====

Dastur

++++

**Obyektga yo'naltirilgan dasturlashda, "try" bloki nima?**

====

#Dastur davomida xatolikni qanday qilib hal qilishga urinish.

====

Klasslarning tavsifi

====

Klasslarning yig'indisi

====

Obyektning yo`naltiriladigan funksiyasi

++++

**C++ tilida "catch" bloki nima uchun ishlatiladi?**

====

# Xatoni hal qilish uchun

====

Klasslarning tavsifi

====

Klasslarning yig'indisi

====

Obyektning yonaltiriladigan funksiyasi

++++

**Obyektga yo'naltirilgan dasturlashda, "throw" operatori qanday ishlatiladi?**

====

#Xatoni qanday hal qilish uchun ishlatiladi.

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

====

Funksiyalarning ichida ishlatiladi

====

Klasslarning tavsifi

++++

**C++ tilida "RAII" nimani ifodalaydi?**

====

#Resurslarni obyektga alohida yaratish

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

====

Funksiyalarning ichida ishlatiladi

====

Klasslarning tavsifi

++++

**Obyektga yo'naltirilgan dasturlashda, "template specialization" nima?**

====

#Ma'lum bir tarkibdagi obyektlarning umumiy tavsifi

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

====

Funksiyalarning ichida ishlatiladi

====

Klasslarning tavsifi

++++

**C++ tilida "STL" nima?**

====

#Standard Template Library

====

Standart Turing Language

====

Standard Type Library

====

Static Typing Language

++++

**Obyektga yo'naltirilgan dasturlashda, "this" kalit so'zi nimani ifodalaydi?**

====

#Obyektni yaratish uchun ishlatiladi

====

Klassning nomini ifodalaydi

====

Klassning doimiy o'zgaruvchisini ifodalaydi

====

Klassning funksiyasini ifodalaydi

++++

**C++ tilida "copy constructor" nima?**

====

#Boshqa obyektning nusxasini yaratish uchun ishlatiladi

====

Bitta obyektning barcha xususiyatlari va funksiyalari

====

Klassda boshqa funksiya bilan o'zlashtirilgan funksiya

====

Klasslarning yig'indisi

++++

**Obyektga yo'naltirilgan dasturlashda, "operator overloading" qanday yoziladi?**

====

#Operator funksiyasi nomi orqali

====

Klassning nomi orqali

====

Konstruktor funksiyasi nomi orqali

====

Barcha funksiyalar uchun

++++

**C++ tilida "friend class" nima?**

====

#Boshqa klassga o'z xususiyatlarini ko'rsatish

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Bitta obyektning barcha xususiyatlari va funksiyalari

++++

**Obyektga yo'naltirilgan dasturlashda, "multiple inheritance" qanday amalga oshiriladi?**

====

#Bitta klassda ikkita yoki undan ko'p bazaviy klasslar

====

Bitta klassdan boshqa klass yaratilishi

====

Klass obyektlarini yaratish paytida chaqiriladigan funksiya

====

Klasslarning yig'indisi

++++

**C++ tilida "const" funksiyasi nima?**

====

# Konstanta o'zgaruvchilarini qo'llab-quvvatlash

====

Konstanta o'zgaruvchisi

====

Konstruktor funksiyasi

====

Obyektning yo`naltiriladigan funksiyasi

++++

**Obyektga yo'naltirilgan dasturlashda, "operator ++" nimani ifodalaydi?**

====

#Obyektga 1 qo'shish operatori

====

Klasslarning tavsifi

====

Obyektning yo`naltiriladigan funksiyasi

====

Operatorlarning klassga xos xususiyatlarini qo'shish

++++

**C++ dasturlash tilida "copy assignment operator" qanday yoziladi?**

====

#Operator funksiyasi nomi orqali

====

Klassning tavsifi

====

Konstruktor funksiyasi nomi orqali

====

Doimiy o'zgaruvchi, klass obyektlariga bog'liq emas

++++

**Obyektga yo'naltirilgan dasturlashda, "overloaded operator" nimani anglatadi?**

====

# Operatorlarning klassga xos xususiyatlarini qo'shish

====

Klassning tavsifi

====

Operator funksiyasi nomi

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

++++

**C++ tilida "exception handling" nimani anglatadi?**

====

#Dastur davomida xatolikni qanday qilib hal qilish

====

Klasslarning tavsifi

====

Klasslarning yig'indisi

====

Obyektning yo`naltiriladigan funksiyasi

++++

**C++ dasturlash tili qanday paradigma(lar)ga ega?**

====

#Protsedural va Obyektga yo`naltirilgan

====

Funksional va Logikaviy

====

Aksioni va Jadvalli

====

Barcha variantlar to'g`ri

++++

**Obyektga yo'naltirilgan dasturlashda, "klas" nimani anglatadi?**

====

# Bitta yoki bir nechta bir xil tarkibdagi obyektlarning umumiy tavsifi

====

Dastur strukturasini ifodalovchi narsa

====

Funksiyalar to'plami

====

Obyektlar to'plami

51……………………………………………………………….

++++

**Obyektga yo'naltirilgan dasturlashda, "inkapsulyatsiya" nimani ifodalaydi?**

====

# Bitta obyektning barcha xususiyatlari va funksiyalari

====

Klasning umumiy tavsifi

====

Klassizatsiya paytida yaratilgan obyektlar

====

Klasslarning yig'indisi

++++

**C++ tilida "polimorfizm" nimani ifodalaydi?**

====

#Bitta funksiya nomi orqali ikkita yoki undan ko'p funksiyani chaqirish

====

Bitta obyektning barcha xususiyatlari va funksiyalari

====

Klassizatsiya paytida yaratilgan obyektlar

====

Klasslarning yig'indisi

++++

**Obyektga yo'naltirilgan dasturlashda, "klas ierarxiyasi" nimani anglatadi?**

====

# Klaslarning tavsifi

====

Klasslar yig'indisi

====

Klasslarning qatorboshi

====

Bitta klassdan boshqa klass yaratilishi

++++

**C++ tilida "abstract klass" nimani anglatadi?**

====

# Bitta xususiyat va funksiyalarni umumlashtirgan klass

====

Obyektga yo'naltirilgan dasturlash paradigmasi

====

Klass ierarxiyasining eng yuqori klassi

====

Bitta funksiya nomi orqali ikkita yoki undan ko'p funksiyani chaqirish

++++

**Obyektga yo'naltirilgan dasturlashda, "virtual funksiya" nima?**

====

#Klassda boshqa funksiya bilan o'zlashtirilgan funksiya

====

Bitta obyektning barcha xususiyatlari va funksiyalari

====

Klassizatsiya paytida yaratilgan obyektlar

====

Klasslarning yig'indisi

++++

**C++ dasturlash tilida "operator overloading" nimani anglatadi?**

====

#Operatorlarning klassga xos xususiyatlarini qo'shish

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Bitta funksiya nomi orqali ikkita yoki undan ko'p funksiyani chaqirish

++++

**Obyektga yo'naltirilgan dasturlashda, "konstruktor" nima?**

====

# Klass obyektini yaratish paytida chaqiriladigan funksiya

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Operatorlarning klassga xos xususiyatlarini qo'shish

++++

**C++ tilida "destruktor" nima?**

====

#Klass obyektining yozib bo'lgan xotirasini o'chirish uchun chaqiriladigan funksiya

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

++++

**Obyektga yo'naltirilgan dasturlashda, "friend funksiya" nima?**

====

#Klassdan tashqari funksiya, ammo unga xos xususiyatlarga ega

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

++++

**C++ tilida "namespace" nima?**

====

#nomlangan obyektlar va funksiyalar uchun ma'lum bir joy

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Klassdan tashqari funksiya, ammo unga xos xususiyatlarga ega

++++

**Obyektga yo'naltirilgan dasturlashda, "template" nima?**

====

# Ma'lum bir tarkibdagi obyektlarning umumiy tavsifi

====

Klassning tavsifi

====

Klasslarning yig'indisi

====

Klass obyektini yaratish paytida chaqiriladigan funksiya

++++

**C++ dasturlash tilida "const" kalit so'zi nimani ifodalaydi?**

====

#Konstanta o'zgaruvchisi

====

Klasslarning tavsifi

====

Konstruktor funksiyasi

====

Obyektning yo`naltiriladigan funksiyasi

++++

**Obyektga yo'naltirilgan dasturlashda, "static" o'zgaruvchi nima?**

====

#Doimiy o'zgaruvchi, klass obyektlariga bog'liq emas

====

Klasslarning tavsifi

====

Konstruktor funksiyasi

====

Obyektning yo`naltiriladigan funksiyasi

++++

**Obyektga yo'naltirilgan dasturlash (OOP) nima?**

====

#Klasslar va obyektlar orqali dasturlash usuli

====

Oddiy dasturlash usuli

====

Dasturlashning yaroqlilik bosqichi

====

Ma'lumotlar bazasi tuzish usuli

++++

**C++ tilida klass qanday yaratiladi?**

====

#class MyClass

====

new Class MyClass

====

define MyClass as class

====

MyClass = new class

++++

**Obyekt nima?**

====

#Klassning bir mazmuni

====

Ma'lumotlar tuzilmasi

====

Klassning nusxasi

====

Dastur tizimini boshqarish

++++

**C++ tilida "constructor" nima uchun ishlatiladi?**

====

# Klass obyektini yaratish uchun

====

Klassning nomini ifodalash uchun

====

Klassni o'chirish uchun

====

Klassni tanlash uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "destructor" nima uchun ishlatiladi?**

====

#Klassni o'chirish uchun

====

Klass obyektini yaratish uchun

====

Klassning nomini ifodalash uchun

====

Klassni tanlash uchun

++++

**C++ tilida "member functions" nima?**

====

#Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Klass obyektini yaratish uchun ishlatiladi

====

Klassni o'chirish uchun ishlatiladi

====

Klassni tanlash uchun ishlatiladi

++++

**Obyektga yo'naltirilgan dasturlashda, "encapsulation" nima?**

====

#Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Klass obyektini yaratish uchun ishlatiladi

====

Klassni o'chirish uchun ishlatiladi

====

Klassni tanlash uchun ishlatiladi

++++

**C++ tilida "inheritance" (vorislik) nima?**

====

#Bitta klass xususiyatlarini boshqa klassga uzatish

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Klass obyektini yaratish uchun ishlatiladi

====

Klassni o'chirish uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "polymorfizm" nima?**

====

#Bitta klassning barcha xususiyatlari va funksiyalari

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Bitta klass xususiyatlarini boshqa klassga uzatish

====

Klass obyektini yaratish uchun

++++

**C++ tilida "operator overloading" nima?**

====

#Operatorlarni klassga xos xususiyatlar bilan qo'shish

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Bitta klass xususiyatlarini boshqa klassga uzatish

====

Klass obyektini yaratish

++++

**Obyektga yo'naltirilgan dasturlashda, "abstract class" nima?**

====

#Klassning barcha funksiyalari "pure virtual" funksiyalardan iborat

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Bitta klass xususiyatlarini boshqa klassga uzatish

====

Klass obyektini yaratish uchun

++++

**C++ tilida "friend function" nima?**

====

#Boshqa klassning maxsus funksiyasi

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Klassning barcha xususiyatlari va funksiyalari

====

Klass obyektini yaratish uchun

76………………………………………………………………….

++++

**Obyektga yo'naltirilgan dasturlashda, "namespace" nima?**

====

#Nomlashni belgilaydigan belgi

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Boshqa klassning maxsus funksiyasi

====

Klass obyektini yaratish uchun

++++

**C++ tilida "template" nima?**

====

# Parametrlar bilan ta’minlangan umumiy funksiya yoki klass

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Boshqa klassning maxsus funksiyasi

====

Klass obyektini yaratish uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "STL" nima?**

====

#Standart Vorislik Til

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Parametrlar bilan ta’minlangan umumiy funksiya yoki klass

====

Klass obyektini yaratish uchun

++++

**C++ tilida "const" nimani anglatadi?**

====

#Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Klassning a'zolari sifatida ta’minlangan funksiyalar

====

Parametrlar bilan ta’minlangan umumiy funksiya yoki klass

====

Klass obyektini yaratish uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "namespace" nima uchun ishlatiladi?**

====

# Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Xatoni qanday hal qilish uchun ishlatiladi

====

Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**Sinf obyektini yaratish uchun qaysi funksiya ishlatiladi?**

====

#constructor().

====

initialize()

====

createObject()

====

newObject()

++++

**C++ tilida sinfning ichki ma'lumotlarini himoya qilish uchun qaysi kalit so'z ishlatiladi?**

====

#private.

====

Protected

====

Secure

====

hidden

++++

**Sinfda ma'lumotlarni umumiy (public) qilish uchun qaysi kalit so'z ishlatiladi?**

====

#public.

====

Common

====

Shared

====

accessible

++++

**C++ tilida sinf ichidagi funksiyalarni e'lon qilish uchun qaysi qism ishlatiladi?**

**====**

#methods.

====

Functions

====

Procedures

====

Actions

++++

**C++ tilida bir sinfdagi obyektning o'zgaruvchilariga qaysi funksiya orqali murojaat qilinadi?**

====

#getters.

====

Accessors

====

Modifiers

====

setters

++++

**Sinfda ma'lumotlarni e'lon qilish uchun qaysi qism ishlatiladi?**

====

#attributes.

====

Data

====

Information

====

variables

++++

**C++ tilida sinfning ichki ma'lumotlarini boshqa sinfdan foydalanish uchun qaysi kalit so'z ishlatiladi?**

====

#friend.

====

Share

====

Access

====

use

++++

**Sinfda ichki ma'lumotlarni yoritish uchun qaysi funksiya ishlatiladi?**

====

#show().

====

display()

====

print()

====

present()

++++

**C++ tilida sinfda ichki ma'lumotlarni boshqarish uchun qaysi kalit so'z ishlatiladi?**

**====**

#private.

====

Secure

====

Protected

====

hidden

++++

**C++ tilida sinf ichidagi funksiyalar qanday turdagi funksiyalardir?**

====

#public.

====

Private

====

Protected

====

static

++++

**Sinfda ma'lumotlarni o'zgartirish uchun qaysi funksiya ishlatiladi?**

====

#set().

====

get()

====

change()

====

modify()

++++

**Sinfda ichki ma'lumotlarni boshqarish uchun qaysi funksiya ishlatiladi?**

====

#set().

====

get()

====

change()

====

modify()

++++

**C++ tilida sinf ichidagi funksiyalarni boshqa sinfdan foydalanish uchun qaysi kalit so'z ishlatiladi?**

====

#friend.

====

Share

====

Access

====

use

++++

**Sinf ichidagi ichki ma'lumotlarni boshqarish uchun qaysi funksiya ishlatiladi?**

====

#set().

====

get()

====

change()

====

modify()

++++

**Sinfda ichki ma'lumotlarni o'zgartirish uchun qaysi funksiya ishlatiladi?**

====

#set().

====

get()

====

change()

====

modify()

++++

**C++ tilida sinf ichidagi funksiyalarning maqsadi nima?**

====

#Obyektga yo'naltirilgan dasturlash uchun.

====

Ma'lumotlarni to'plash

====

Matematik amallarni bajarish

====

Fayllar bilan ishlash

++++

**C++ tilida obyektga kirish (ma’lumot kiritish) qanday ishlatiladi?**

====

#cin.

====

Read

====

Access

====

Input

++++

**C++ tilida ma'lumot olish uchun qaysi operator ishlatiladi?**

====

# >>.

====

<<

====

=>

====

<:

++++

**C++ tilida obyektga ma'lumot kirish uchun qanday operator ishlatiladi?**

====

#->.

====

<-

====

=>

====

<:

++++

**C++ tilida obyektga kirish uchun qaysi standart funksiya ishlatiladi?**

====

#cin().

====

get()

====

input()

====

read()

101…………………………………………………………………..

++++

**C++ dasturlash tilida "meros" nima ifodalaydi?**

====

#Meros dasturlash tilidagi obyektga yo'naltirilgan dasturlash.

====

Meros dasturlash tilidagi o'yinlar

====

Meros dasturlash tilidagi obyektlar

====

Meros kengaytirilgan dasturlash tilidagi funksiyalar

====

++++

**Meros dasturlashning asosiy mablag‘i nima?**

====

#Obyektlar.

====

Funksiyalar

====

Massivlar

====

Operatorlar

++++

**Meros dasturlash tilida obyektlar qanday yaratiladi?**

====

#new().

====

define()

====

object()

====

create()

++++

**Meros dasturlashda obyektning xususiyatlarini aniqlash uchun qaysi metod ishlatiladi?**

====

#set().

====

define()

====

initialize()

====

create()

++++

**C++ tilida meros dasturlashni amalga oshiruvchi kalit so‘z qanday?**

====

#class.

====

Function

====

Object

====

main

++++

**Abstrakt sinf nima uchun ishlatiladi?**

====

#Xususiyatlarni yig'ib olish uchun.

====

Obyekt yaratish uchun

====

Ko'p qo'llanuvchili dasturlarni yaratish uchun

====

Qo'shimcha funksiyalar qo'shish uchun

++++

**Abstrakt sinflar necha xil sinfda ishlatilishi mumkin?**

====

#Ko'p xil sinflarda.

====

Faqat bir xil sinfda

====

Ular umumiy sinflarda ishlatiladi

====

Ular faqat turli funksiyalarda ishlatiladi

++++

**C++ tilida abstrakt sinfda qanday metodlar ishlatiladi?**

====

#Ularning ichida barchasi.

====

Ularning ichida birinchilari

====

Ularning ichida oxirigilar

====

Ularning ichida hech qaysi emas

++++

**C++ tilida abstrakt sinflar qanday xususiyatlar uchun ishlatiladi?**

====

#Ularning ichida oxirigilar.

====

Ularning ichida barchasi

====

Ularning ichida hech qaysi emas

====

Ularning ichida birinchilari

++++

**C++ tilida abstrakt sinflar qanday yaratiladi?**

====

#abstract class.

====

virtual class

====

Aabstract

====

Virtual

++++

**C++ tilida abstrakt sinfda qanday xususiyatlar ishlatiladi?**

====

#Ularning ichida oxirigilar.

====

Ularning ichida barchasi

====

Ularning ichida hech qaysi emas

====

Ularning ichida birinchilari

++++

**Abstrakt sinfda necha xil sinfda ishlatilishi mumkin?**

====

#ko'p xil sinflarda

====

Ular umumiy sinflarda ishlatiladi

====

Ular faqat turli funksiyalarda ishlatiladi

====

Faqat bir xil sinfda

++++

**Sinf pointeri nima uchun ishlatiladi?**

====

#ob'ekt yaratish uchun

====

Xususiyatni ishlatish uchun

====

Funksiyani ishlatish uchun

====

Xususiyatni e'lon qilish uchun

++++

**Sinf pointeri (class pointer) nima uchun ishlatiladi?**

====

#ob'ekt yaratish uchun

====

Xususiyatni ishlatish uchun

====

Funksiyani ishlatish uchun

====

Xususiyatni e'lon qilish uchun

++++

**C++ tilida sinf pointerini yaratish uchun qanday sintaksis ishlatiladi?**

====

#MyClass \*pointer = &MyClass;

====

myClass.pointer class;

====

classPointer MyClass;

====

class MyClass = new class;

++++

**Sinfda pointer qanday ishlatiladi?**

====

#. Operatori bilan

====

barchasi to'g'ri

====

> operatori bilan

====

< operatori bilan

++++

**Sinf pointeri o'zgaruvchilarga bog'liq qanday metodlarga ega bo'lishi mumkin?**

====

#faqat public metodlarga

====

Faqat private metodlarga

====

Xech narsaga bog'liq emas

====

Barchasi to'g'ri

++++

**Sinf shablonlari (class templates) nima uchun ishlatiladi?**

====

#turli ma'lumot turlariga qo'shimcha imkoniyat yaratish uchun

====

Qo'shimcha xususiyatlarni qo'shish uchun

====

Yagona sinflarni yaratish uchun

====

Obyektlarni yaratish uchun

++++

**Sinf shablonlarini yaratishda qanday belgilar ishlatiladi?**

====

#< >

====

{ }

====

[ ]

====

( )

++++

**Sinf shabloni yaratish uchun C++-ning sintaksisi qanday?**

====

#template <T> class MyClass { };

====

Template <class T> class MyClass;

====

Class MyClass template <T>;

====

Class MyClass <T> { };

++++

**Sinf shablonlari orqali yaratilgan sinf obyektlarini qanday yaratish mumkin?**

====

#MyClass<int> obj;

====

myClass obj<int>;

====

myClass obj = new MyClass<int>;

====

myClass<int> obj();

++++

**Sinf shablonida foydalanish uchun kerak bo'lgan ma'lumot turlari qanday belgilanadi?**

====

#Typename

====

class

====

type

====

template

++++

**Sinf shablonlarida qanday ko'rsatkichlar ishlatiladi?**

#,

====

^

====

&

====

%

++++

**Sinf shablonlari qanday asosiy sinf turlarini ishlatish imkoniyatini beradi?**

====

#typename

====

T

====

Template

====

Class

++++

**Sinf shablonlari qanday maqsadga muvofiq yaratiladi?**

====

#ko'p xil ma'lumot turlariga qo'shimcha imkoniyat yaratish uchun

====

Operatorlar orqali ishlov berish uchun

====

Obyektlarni yaratish uchun

====

Mantiqiy xatolar aniqlash uchun

126………………………………………………………………………….

++++

**Sinf shablonlarida qanday ma'lumot turlari yaratish mumkin?**

====

#turli ma'lumot turlari

====

Faqat ob'ektlar

====

Faqat sonlar

====

Faqat matnlar

++++

**Sinf shablonida ma'lum bir xususiyatni ishlatish uchun qanday sintaksis ishlatiladi?**

**====**

#T->property

====

t.property()

====

t::property

====

t.property

++++

**Sinf shablonida xususiyatlarni ishlatish uchun qanday sintaksis ishlatiladi?**

====

#T::property

====

T.property()

====

T.property

====

T->property

++++

**Sinf shablonida foydalanuvchi kiritgan ma'lumotlarni qanday qabul qilish mumkin?**

====

#cin

====

Get

====

Input

====

User\_input

++++

**Sinf shablonlarida qanday xususiyatlar ishlatiladi?**

====

#T.property

====

T->property

====

T::property

====

T.property()

++++

**Sinf shablonlarida foydalanuvchi kiritgan ma'lumotlarni qanday qabul qilish mumkin?**

====

#cin

====

Get

====

Input

====

User\_input

++++

**Sinf shablonlarida qanday yordam orqali ma'lumotlarni qaytarish mumkin?**

====

#result

====

Return

====

Output

====

Get

++++

**Sinf shablonlari qanday metodlarni ishlatish imkoniyatini beradi?**

====

#hamma turdagi metodlar

====

Xech qanday metodlar emas

====

Faqat public metodlar

====

Faqat private metodlar

++++

**Sinf shablonlarida qanday metodlarni ishlatish uchun qanday sintaksis ishlatiladi?**

====

#template <class T> void myMethod();

====

Void myMethod<T>();

====

Void myMethod();

====

Template void myMethod<T>();

++++

**Sinf shablonlarida xususiyatlarni ishlatish uchun qanday sintaksis ishlatiladi?**

====

#T.property

====

T->property

====

T::property

====

T.property()

++++

**Parallel dasturlash nima?**

====

#bir nechta dasturlarni bir vaqtda bajarish

====

Dastur kodini bitta joyda saqlash

====

Dasturda parallel operator ishlatish

====

Bitta dasturni bir nechta o'zgaruvchilar bilan ishlatish

++++

**Paralel dasturlashda qanday qilib xodimlar birlashtiriladi?**

====

#mutex

====

Thread

====

Semaphore

====

Pointer

++++

**Dasturlashda deadlock nima?**

====

#bitta xodimning boshqa xodimni kutishi

====

Xodimlar birlashtirilmaganligi

====

Dasturni tizimi tushirish

====

Dasturda qo'llanmalar qo'shib, uni yegishish

++++

**Thread va Process orasidagi asosiy farq nima?**

====

#process bitta dastur, Thread esa dasturning qisqacha vazifasi

====

Thread yordamchi, Process bitta dastur

====

Thread o'zgaruvchilar bilan ishlaydi, Process esa hech qanday o'zgaruvchi ishlata olmaydi

====

Thread dastur bajarish uchun yordamchi protsessdir

++++

**Thread-safe kodi qanday taniladi?**

====

#kod xodimlar tomonidan o'zgartirila oladi

====

Kod parallel dasturlashga moslashtirilgan

====

Kod yozuvchisi uchun xavfsiz

====

Bir vaqtda faqat bitta xodim ishlaydi

++++

**Obyektga yo'naltirilgan dasturlashda, "friend" klass nima uchun ishlatiladi?**

====

#Bitta klassning boshqa klassdagi maxsus funksiyasi.

====

Xatoni qanday hal qilish uchun ishlatiladi

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

**====**

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

**++++**

**C++ tilida "this" kalit so'zi nimani anglatadi?**

====

#Hozirgi obyektni ifodalaydi.

====

Xatoni qanday hal qilish uchun ishlatiladi

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

**++++**

**Obyektga yo'naltirilgan dasturlashda, "static" kalit so'zi nima uchun ishlatiladi?**

====

#Statik a'zoni (statik metodi) ifodalaydi.

====

Xatoni qanday hal qilish uchun ishlatiladi

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**C++ tilida "const" funksiya nima uchun ishlatiladi?**

====

# Obyekt o'zgartirilmaydiganligini ifodalovchi belgi.

====

Xatoni qanday hal qilish uchun ishlatiladi

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Statik a'zoni (statik metodi) ifodalaydi

**++++**

**Obyektga yo'naltirilgan dasturlashda, "polymorphism" qanday shakllanadi?**

====

#Obyektning yo`naltiriladigan funksiyalari orqali.

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Statik a'zoni (statik metodi) ifodalaydi

**====**

**++++**

**C++ tilida "virtual" funksiya nima uchun ishlatiladi?**

**====**

#Polimorfizmni qo'llash uchun.

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Statik a'zoni (statik metodi) ifodalaydi

**++++**

**Obyektga yo'naltirilgan dasturlashda, "abstract class" nima?**

====

#Klassning barcha funksiyalari "pure virtual" funksiyalardan iborat.

====

Polimorfizmni qo'llash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Faqat bir marta ishlatiladigan sinf

**++++**

**C++ tilida "inline" funksiya nima uchun ishlatiladi?**

====

#Klassning funksiyasini joylashtirgan joyda bitta blokni ko'chirish uchun.

====

Polimorfizmni qo'llash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Faqat bir marta ishlatiladigan sinf

====

++++

**C++ tilida "try-catch" operatorlarining vazifasi nima?**

====

#Xatoni aniqlash .

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Nomlashni belgilaydigan belgi

====

Klassning funksiyasini joylashtirgan joyda bitta blokni ko'chirish uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "namespace" (nomlar o'qitish) nima uchun ishlatiladi?**

====

#Ma'lumotlarni yig'ish uchun.

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Nomlashni belgilaydigan belgi

151…………………………………………………………………………..

**++++**

**C++ tilida "template" nima uchun ishlatiladi?**

====

#Umumiy funksiyalarni parametrlar bilan ta’minlash uchun.

====

Polimorfizmni qo'llash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Nomlashni belgilaydigan belgi

++++

**Obyektga yo'naltirilgan dasturlashda, "STL" (Standart Vorislik Til) nima?**

====

# Standart vorislik kutubxonasi.

====

Polimorfizmni qo'llash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Umumiy funksiyalarni parametrlar bilan ta’minlash uchun

++++

**Obyektga yo'naltirilgan dasturlashda, "dynamic\_cast" qanday ishlaydi?**

====

# Ishlatiladigan obyektni turini boshqa turga o'zgartirish.

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Standart vorislik kutubxonasi

++++

**Obyektga yo'naltirilgan dasturlashda, "friend" funksiya nima uchun ishlatiladi?**

====

#Bitta klassning boshqa klassdagi maxsus funksiyasi.

====

urli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyekt o'zgartirilmaydiganligini ifodalovchi belgi

====

Standart vorislik kutubxonasi

++++

**Abstrak klass nima uchun ishlatiladi?**

====

#Obyektni asosiy tushuncha va xususiyatlaridan boshqa narsalarni e'lon qilmaslik.

====

Obyektni faqat bir marta ishlatish

====

Barcha detallarni ko'rsatish

====

Detallarning yig'indisini ko'rsatish

++++

**UML nima uchun ishlatiladi?**

====

#Dasturlashda obyektlarni modellash uchun grafik tili.

====

Standart vorislik kutubxonasi

====

Funksiyalarni parametrlar bilan ta’minlash uchun

====

Ma'lumotlar turlarini bir-biriga solishtirish uchun

++++

**UML ichidagi "Class Diagram" nima uchun ishlatiladi?**

====

#Tizimda klasslar va ularning o'zaro aloqalari tasvirlanadi.

====

Xatoni aniqlash uchun

====

Ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Klassning funksiyasini joylashtirgan joyda bitta blokni ko'chirish uchun

++++

**UML ichidagi "Use Case Diagram" nima uchun ishlatiladi?**

====

#Foydalanuvchi kutilayotgan harakatlarni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Sequence Diagram" nima uchun ishlatiladi?**

====

#Foydalanuvchi kutilayotgan harakatlarni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "State Diagram" nima uchun ishlatiladi?**

====

#Holat o'zgarishlarini tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Component Diagram" nima uchun ishlatiladi?**

====

#Komponentlar va ularning o'zaro bog'lanishlari tasvirlanadi.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

**++++**

**UML ichidagi "Deployment Diagram" nima uchun ishlatiladi?**

====

#Tizimning fizikaviy qurilishi va ularning aloqalari tasvirlanadi.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

**++++**

**UML ichidagi "Object Diagram" nima uchun ishlatiladi?**

====

#Obyektlarning ma'lumotlarini tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Package Diagram" nima uchun ishlatiladi?**

====

#Paketlarni va ularning bog'lanishlarini tasvirlanadi.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Composite Structure Diagram" nima uchun ishlatiladi?**

====

#Komponentlarning ichki tuzilishi va ularning aloqalari tasvirlanadi.

=====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Profile Diagram" nima uchun ishlatiladi?**

====

#Profillarni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

**++++**

**UML ichidagi "Interaction Overview Diagram" nima uchun ishlatiladi?**

====

# Bir qancha turdagi interaksiyalarni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Timing Diagram" nima uchun ishlatiladi?**

====

#Vaqtni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Communication Diagram" nima uchun ishlatiladi?**

====

#Obyektlar orasidagi axborot almashishni tasvirlash uchun.

====

Xatoni aniqlash uchun

====

Sistemada klasslar va ularning o'zaro aloqalari tasvirlanadi

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Association Class" nima uchun ishlatiladi?**

====

#Bitta klassning boshqa klassdagi maxsus funksiyasi.

====

Xatoni aniqlash uchun

====

Obyektlar orasidagi axborot almashishni tasvirlash uchun

====

Ishlatiladigan obyektni turini boshqa turga o'zgartirish

++++

**UML ichidagi "Object Constraint Language (OCL)" nima uchun ishlatiladi?**

====

#Ma'lumotlar turlarini bir-biriga solishtirish uchun.

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyektlarning ma'lumotlarini tasvirlash uchun

++++

**UML ichidagi "Diagram Frames" nima uchun ishlatiladi?**

====

#har bir diagrammadagi elementlarni guruhlash uchun

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyektlarning ma'lumotlarini tasvirlash uchun

++++

**UML ichidagi "Derived Class" nima uchun ishlatiladi?**

====

#boshqa klassdan olingan klass

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Ma'lumotlar turlarini bir-biriga solishtirish uchun

++++

**UML ichidagi "Link" nima uchun ishlatiladi?**

====

#obyektlarning aloqalarini tasvirlash uchun

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyektlarning ma'lumotlarini tasvirlash uchun

++++

**UML ichidagi "Node" nima uchun ishlatiladi?**

====

#tizimda faol bo'lmagan obyekt

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyektlarning aloqalarini tasvirlash uchun

176………………………………………………………………………………………..

++++

**UML ichidagi "Package" nima uchun ishlatiladi?**

====

#komponentlarni guruhlash uchun

====

Xatoni aniqlash uchun

====

Turli xil ma'lumotlar turlarini bir-biriga solishtirish uchun

====

Obyektlarning ma'lumotlarini tasvirlash uchun

++++

**Obyektga yo'naltirilgan dasturlashning asosiy tushunchasi nima?**

====

#dastur tuzilmasi bo'yicha tashkil etilgan

====

Qat'iy kodi

===

Ehtimol nazorat

====

Kodning bir xil bo'lishi

++++

**Obyektga yo'naltirilgan dasturlashda "encapsulation" nima uchun ishlatiladi?**

====

#obyektlar orasidagi aloqani chegaralash uchun

====

Kodni optimallashtirish uchun

====

Obyektning turini o'zgartirish uchun

====

Tizimni yaxshi tuzilgan va tahlil qilingan qismga bo'lish uchun

++++

**Obyektga yo'naltirilgan dasturlashda "inheritance" nima uchun ishlatiladi?**

====

#bir sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Obyektlar orasidagi aloqani chegaralash uchun

====

Tashqi interfeysslarni ulashish uchun

====

Bitta obyektni boshqasi orqali ishlatish uchun

++++

**Obyektga yo'naltirilgan dasturlashda "polymorphism" (polimorfizm) nima uchun ishlatiladi?**

====

#turli Obyektlar orasidagi aloqani chegaralash uchun

====

Kodni optimallashtirish uchun

====

Bitta obyektning turini o'zgartirish uchun

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

++++

**Obyektga yo'naltirilgan dasturlashda "polymorphism" (polimorfizm) qanday turlarga bo'linadi?**

====

#sinflarning o'zgartiruvchi polimorfizmi

====

Faqat o'zgaruvchilarni o'zgartiruvchi polimorfizm

====

Sinflarning o'zgartirilmaydigan polimorfizmi

====

Turli obyektlar orasidagi aloqani chegaralash uchun

++++

**Obyektga yo'naltirilgan dasturlashda "class" (sinf) nima?**

====

#bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

====

Bitta obyektni boshqasi orqali ishlatish uchun

++++

**Obyektga yo'naltirilgan dasturlashda "constructor" (konstruktor) nima uchun ishlatiladi?**

====

#obyekt yaratilganda avtomatik bajariladigan boshlang'ich funksiya

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**Obyektga yo'naltirilgan dasturlashda "destructor" nima uchun ishlatiladi?**

====

#obyekt yo'qolishi yoki yozuvni yopish uchun

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**UML nima?**

====

#unified Modeling Language.

====

Universal Markup Language

====

Universal Modeling Language

====

Unified Markup Language

++++

**UML da "Class" elementi nima uchun ishlatiladi?**

====

#bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

====

Bitta obyektni boshqasi orqali ishlatish uchun

++++

**UML da "Association" (assotsiatsiya) nima?**

====

#obyektlar o'rtasidagi aloqani ko'rsatadi

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**UML da "Inheritance" (muruvvat) nima?**

====

#bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**UML da "Object" (obyekt) nima?**

====

#bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

====

Bitta obyektni boshqasi orqali ishlatish uchun

++++

**UML da "Association" (assotsiatsiya) uchun necha turi bor?**

====

#2

====

3

====

4

====

1

++++

**UML da "Aggregation" (agregatsiya) nima?**

====

#turli Obyektlar orasidagi aloqani chegaralash uchun

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Obyektning turini o'zgartirish uchun

====

Bitta obyektni boshqasi orqali ishlatish uchun

++++

**UML da "Composition" (kompozitsiya) nima?**

====

#bir obyekt boshqasiga tegishli bo'lib, bir xil hayotiy o'zgaruvchilarni o'z ichiga oladi

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**UML da "Dependency" (bog'lanish) nima?**

**====**

**#**Bitta obyektni boshqasi orqali ishlatish uchun

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

**++++**

**UML da "Association" (assotsiatsiya) va "Dependency" (bog'lanish) orasidagi asosiy farq nima?**

====

#assotsiatsiya o'zaro aloqador sinflarni bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

====

Assotsiatsiya o'zaro aloqador sinflarni bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

====

Assotsiatsiya bitta obyektning boshqasi orqali ishlatilishini bildiradi, bog'lanish esa o'zaro aloqador sinflarni bildiradi

====

Assotsiatsiya bitta obyektning boshqasi orqali ishlatilishini bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

++++

**UML da "Dependency" (bog'lanish) nima?**

====

#bitta obyektni boshqasi orqali ishlatish uchun

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**UML da "Association" (assotsiatsiya) va "Dependency" (bog'lanish) orasidagi asosiy farq nima?**

====

#assotsiatsiya o'zaro aloqador sinflarni bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

====

Assotsiatsiya o'zaro aloqador sinflarni bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

====

Assotsiatsiya bitta obyektning boshqasi orqali ishlatilishini bildiradi, bog'lanish esa o'zaro aloqador sinflarni bildiradi

====

Assotsiatsiya bitta obyektning boshqasi orqali ishlatilishini bildiradi, bog'lanish esa bir sinfni boshqasi orqali ishlatishni ko'rsatadi

++++

**UML da "Interface" (interfeys) nima uchun ishlatiladi?**

====

#bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Bitta sinfga mansub barcha xususiyatlar va funksiyalarni o'z ichiga oladi

====

Turli Obyektlar orasidagi aloqani chegaralash uchun

====

Obyektning turini o'zgartirish uchun

++++

**C++ dasturlash tilida Sinflar nima uchun ishlatiladi?**

====

#obyektga yo'naltirilgan dasturlash uchun

====

Funksiyalar yig'ish uchun

====

Matematik amallarni bajarish uchun

====

Fayllar bilan ishlash uchun

++++

**C++ tilida sinfni tavsiflash uchun qaysi kalit so'z ishlatiladi?**

====

#class

====

Struct

====

Object

====

Type

++++

**Sinfda ma'lumotlarni o'zgartirish uchun qanday funksiya ishlatiladi?**

====

#set()

====

Get()

====

Change()

====

Modify()