**- OOP differences between prototype based and class-based**

In prototype-based OOP, object’s properties and methods are not defined by a class, but are instead determined by the object’s prototype. Inheritance implemented via cloning. (JavaScript)

In class-based OOP, objects are created by creating an instance of the class. The class defines the properties and methods of an object. (Java, C++, C#)

**- differences between expression and statements**

Expression returns a value, it can be used on right-hand side of an assignment statement, passed as an argument to a function etc. (variables, literals, operators)

x+y

Statements performs an action; they do not return a value and they control the flow of a program. (Assignments, control statements, declarations.)

x = x+y

if (x>0) {…}

In summary, the main difference is:  
-expression: no-side effect, has a return value  
-statement: performs an action, side effect, no return value,

-Assignment side effect, no return value

**- Why dynamic lookup is better?**

Dynamic lookup: Ability of PL to determine appropriate method or function to call at runtime, rather than at compile-time.

-**Flexible**: It enables objects to change their behavior at runtime, it can adapt.

-**Polymorphism**: Different objects calls the same method in different ways.

-**Extensibility**: It allows new classes and objects to be added without modifying existing code.

-**Performance**: In some cases, it has a better performance because the program will avoid determining the methods to call at compile-time.

**What are the differences of type inference and type checking?**

Type inference: Programmer does not have to specify the type, compiler or interpreter can deduce/decide them based on the context. (Functional Languages)

Type checking: The compiler or interpreter will check that if types are used correctly, if not, it will result in error.

**How C++ provides ADT and what its difference from module?**

In C++, ADTs are implemented using classes, which define new data types. A module, on the other hand, is a unit of organization for code and data that groups related declarations together.

**Differences of C++ and Java.**

They are both high-level PLs.

C++ is a compiled language, java is primarily interpreted.

Java is OOP language,

C++ support pointers while java does not, it uses references.

C++ has operator overloading, while java does not.

Java has garbage collector, C++ requires manual memory management.

**What are the language evaluation criteria?**

Readability, writability, orthogonality, flexibility, reliability, safety.

**Procedure and function differences**

Function:

* no side effect
* return a value
* Function call: expression

Procedure:

* side effect, executed for it
* no return value / action performed
* Procedure call: statement

In short, Functions are used for returning value, procedures are used for performing an action, and no return value is expected.

**\*\*Statement and expression differences**

Statement: perform actions, (control flow, variable declarations) side effect, no return value, do not have a value

Expression: used for computation, no side effect, has a value

Assignments: used for assigning a value to a variable, assigns the value of an expression to a variable

In short, An assignment statement assigns the value of an expression to a variable, Expressions are used for computation, and statements are used for performing an action.

**What is “Module”? Why do we use it? What is its benefit to ADT?**

a section of code that is added in as a whole or is designed for easy reusability. It is used to group related code together in a single unit, avoiding naming conflict and making the code more manageable. It benefits ADT by providing namespacing, reusability and encapsulations, which makes the code more organized.

**Differences of module and namespace**

They both used to organize and group related code. But in module, the organized code and data can be imported and exported, while in namespace, it only group them together in a organized way and avoid naming conflicts.

**Advantages and disadvantages of Functional Programming**

Advantages: simplicity, modularity, and concurrency.

Disadvantages: verbosity, performance and learning curve.

**What is overloading and polymorphism? What are the differences?**

Overloading is multiple functions with the same name.

Polymorphism is about treating different types as a common type.

**Explain axiom, horn clause, query, and give examples.**

Axiom: A statement that is true: It is sunny.

Horn clause: A logical statement: If it rains, the grass is wet (rain->wet)

Query: Goal: Is 2 a natural number?

**Explain abstract class and interface. What are the differences?**

Abstract classes are a class with at least one abstract method, which is a method without a body. An interface is all abstract methods, without a body, that a class must implement.

**How can you show if a given CFG is ambiguous?**

A CFG is ambiguous if there is at least one string that is generated by the grammar has more than one different parse trees.

**Explain explicit heap-dynamic variables.**

These variables (aka. Dynamic memory allocation) is the process of allocating memory on the heap during runtime. New in C++, malloc in C.

**What are discriminated unions?**

Single variable can hold multiple types of data. Union in C++.

**Why does C have separate compilation?**

Because it allows for the development the larger programs to be divided into smaller, more manageable units and this improves build times.

**What is the difference between a macro and call by reference?**

Macros are used to define a sequence of symbols with another symbols before compilation of the source code, for example, it is used to define a constant value that is be used in the code.

Call by reference is a way to pass arguments in a function. Instead of passing the value of the variable, we pass the reference to the original argument, this allows the function to directly affect the original value.

**What are namespaces?**

Namespaces are used to group parts of the code that is relevant with each other to write a more organized code and reduce the risk of name conflicts.

**What are class variables and methods? Provide examples.**

Class variables are variables that are used among all of the objects of that class. For example:

Class Students{

Private int student\_id;

Private static int student\_count;

…

Class methods are methods that are used among all of the object of the class and is not called on any specific object.

Class Students{

Private int student\_id;

Private static int student\_count;

…

Public static int getTotalStudentNumber(){return student\_count;}

**What are object/instance methods and variables. Provide examples.**

Instance variables are variables that is specific to an instance that is created. It is different on each instances and changing one will not affect the other.

Instance methods are methods that is specific to an instance that is created. It is called through the instance and will use its instances values.

**What is the difference between a class and an object in Smalltalk?**

In smalltalk, everything is an object, and classes act as descriptions of objects.

**How are interfaces realized in C++?**

Interfaces are realized with interface keyword, also, their functions has no body.

**What is an expression?**

Expression is a computation that has a result, return value. There can be one expression or group of expressions. Ex: a + b or (3\*2)/5

**What are “eager” and “lazzy” evaluations?**

Eager evaluation means that the expression will be evaluated as soon as it is encountered, while in lazzy evaluation, it will be evaluated when its needed.

**When would lazzy evaluation be useful?**

It can be useful if high performance is needed.

**What is a “lambda expression”? What is it used for?**

Lambda expressions are short block of code that is used to evaluate some expressions depending on taken parameter, and returns a value. They are similar to methods, but they don’t need a name and can be implemented right in the body of a method.

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**What form do all production rules in a right regular grammar have? Explain the symbols.**

**What are the three main differences between a procedure and a function?**

Return value: functions has, procedures not

Modifies state: functions not, procedures does

Function can be used as an expression

**What is an activation record in a procedural language? How does it get allocated and deallocated?**

**Indicate the type systems for the following languages?**

**Haskell:** Strongly **Prolog:** Weakly

**C:** Weakly  **Java:** Almost Strongly

**What are the differences between an expression and an assignment?**

Expression: no-side effect, has a return value

Assignment= side effect, no return value

**Explain why lazy evaluation is not used when side effect is allowed in an expression or a function?**

When side effects are allowed, the order of evaluation is important and lazy evaluation is not used as it may lead to unexpected or inconsistent behavior.

**Prolog factorial:**

metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Prolog Fibonacci:**metin içeren bir resim

Açıklama otomatik olarak oluşturuldu

Even numbers between:

metin, ekran, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu