365 MIDTERM NOTES

Which language inspired the syntax for most common C-like languages?

- ALGOL

Be able to associate some of the first high-level languages with their primary paradigms

- COME BACK TO THIS ONE

What is syntax?

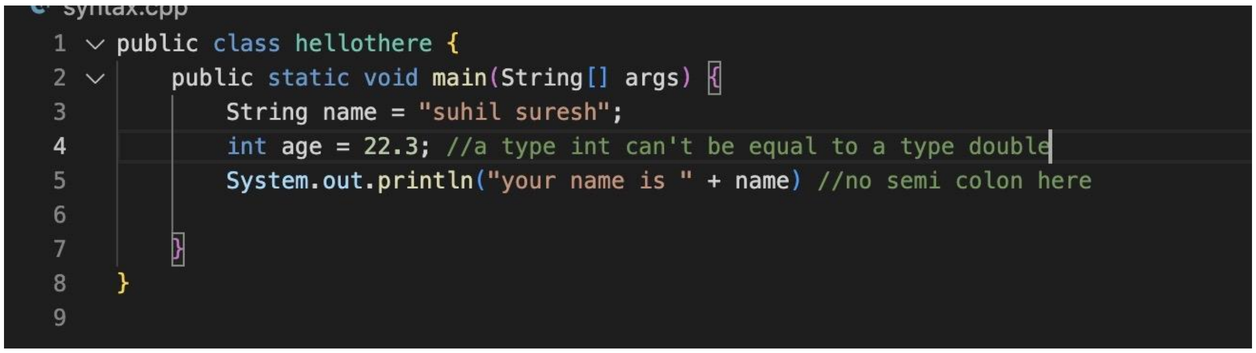
- Syntax is when the code that is written is grammatically correct and has to check whether the program is structurally valid.

What are semantics?

- Semantics are when the program is checked for the meaning rather than grammar concepts.

Given a simple example compiler error, identify whether it is syntactic or semantic.

- Syntax:

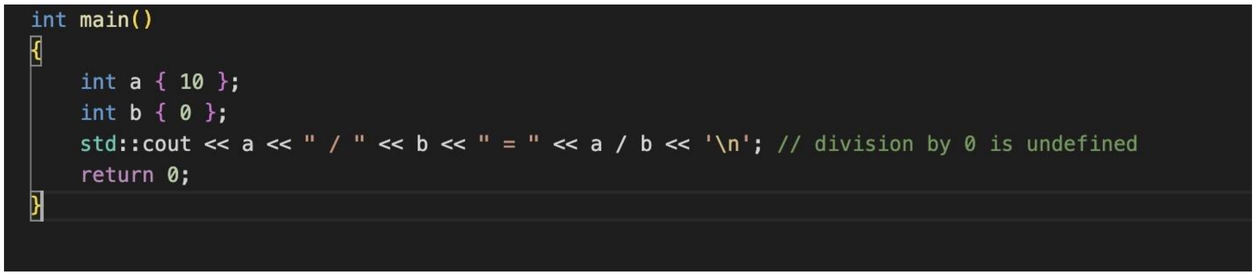


```
syntax.cpp
public class hellothere {
    public static void main(String[] args) {
        String name = "suhil suresh";
        int age = 22.3; //a type int can't be equal to a type double
        System.out.println("your name is " + name) //no semi colon here

    }
}

```

- Semantic:



```
int main()
{
    int a { 10 };
    int b { 0 };
    std::cout << a << " / " << b << " = " << a / b << '\n'; // division by 0 is undefined
    return 0;
}
```

How does grammar relate to syntax?

- Grammar can be used to create language recognizers; describes how to create strings fitting the rules of a formal language.

Basic components of grammar:

- Terminals: symbols from alphabet
- Non-terminals: Name that refers to parts of grammar
- Productions: rules that define replacement of non-terminals with terminals
- Start symbol: describes what a complete program looks like in the language to be parsed (its where the rewriting of a string begins)

What is a type system?

- A set of semantic rules that assign a “type” to different parts of a program and checks for consistency between typed elements

What are the benefits of having a type system in a language?

- Computes size of data elements (int x)
- Relate data to operations (x + 1)
- Enforce consistency and correctness (x = abc) //error: no string and int

What is the difference between a static and dynamic type system?

- **Static**: all type errors are caught by the compiler/type checker before program runs (type can't change)
- **Dynamic**: type errors may happen at runtime (types change as program runs)

Difference between strong and weak type system

- **Strong**: has strict type rules and seeks to catch as many errors as possible during type checking
- **Weak**: more loosely defined and may allow more implicit conversions and/or operations between types

What is the difference between a domain-specific language and a general purpose language?

- Domain: language used for specific purpose or domain app
- General: a widely applicable language that can be used to create a wide variety of computer programs (Python, C++)

How does Turing-completeness relate to programming languages?

- Capable of simulating any Turing machine
- Given enough time and memory, can solve any computational problem

Discuss the differences and tradeoffs between between manual and automatic memory management

- Downsides (Manual)
  - easy to create memory bugs (segmentation violations, mem leaks)
  - easy to create security vulnerabilities (buffer overflow, race conditions)
  - takes a trained programmer to manage memory correctly and efficiently
- Advantages (Automatic)
  - memory that is no longer needed is called garbage
  - runtime performs extra safety checks
  - makes code simpler and easier to write correctly

Describe the major problem with reference counting as a memory-management strategy

- The Cycle Problem: a group of objects may reference each other, creating cyclical references. The reference counting technique will never be able to free a group of objects.

Describe the primary jobs of a garbage collector

- **Garbage detection**: garbage collector automatically detects when an object is no longer needed.
- **Garbage reclamation**: removes it after it realizes that it is no longer needed.

What is a programming language paradigm?

- A classification of PLs based on features and "big ideas"

What is the primary feature that makes a language "imperative"

- Executes commands in a sequence

What is the primary feature that makes a language "procedural"

- When you can define functions

Describe the basic premise of OOP?

- A programming paradigm in which software is organized around the concepts of "objects" containing data and behavior.
    - use and manage interactions between objects to solve tasks
- WHEN SHOULD I USE IT?
    - When the problem you are trying to solve includes many reusable parts.
- WHEN SHOULD I NOT USE IT?
    - When focusing on performance
    - When needing to add concurrency

List and describe each of the three pillars of OOP

- **Inheritance**: ability to define new classes based on a pre-existing class. Classes derived from another class inherit its data members and methods may add additional members. Allows code to be reused. Derived classes can present the same interfaces as a base class.
- **Polymorphism**: An attribute of a type system, many different types (shapes) can share an interface. Using inheritance with virtual/abstract methods, single type can have many shapes, as implemented by each subclass references to subtype obj can be created, passed, and stored, as reference to base class.
- **Encapsulation**: preventing outside access to internal class data/behavior. Ensures that the state of an object is managed solely by that object. Class members are private by default. Members can be shared by using public keywords.

What is the difference between a class and an object?

- Class: classes are used as a template to create an object
- Object: objects are instances of classes

What is polymorphism?

- An attribute of a type system, many different types (shapes) can share an interface. Using inheritance with virtual/abstract methods, single type can have many shapes, as implemented by each subclass references to subtype obj can be created, passed, and stored, as reference to base class.

Describe two forms of polymorphism
- **Subtype**: the ability to use a subclass where a superclass is
- **Parametric**: allows the same piece of code for different types

What does it mean for a class to be abstract?
- It's a class that is designed to be specifically used as a base class. They are classes that contain one or more behaviors or methods.

What is a virtual method and how does it relate to polymorphism?
- A virtual method is a base class member function that you can redefine in a derived class to achieve polymorphism

What is the difference between pass-by-value and pass-by-reference?
- Pass-by-value: you pass an **actual value of variable** into the function
- Pass-by-reference: you pass the **actual variable** into the function

Identify the stages of traditional ahead-of-time compilation
- Does not incur compilation costs at program startup
- Optimization can be more thorough since they only need to be performed once
- Can't take advantage of optimizations that use information only available at runtime.

Explain what a compiler optimization is
- A technique used by compilers to improve the performance of the code generated from the source code.

What does a linker do?
- The linker aggregates object files and makes sure that all external symbols are actually defined somewhere. It is also the program that makes executable files.

Describe the difference between AOT compilation and JIT compilation
- AOT compilation: occurs at build time
- JIT compilation: occurs at runtime

Describe the tradeoffs between interpretation and JIT compilation
- While the interpreted program is being run, the JIT compiler determines the most frequently used code and compiles it to machine code. Interpreters read your high level language and executes what's asked by the program

In general terms, what is a virtual machine?
- An execution environment that allows code for one platform(guest) to run on another(host)

What are some of the things that a high level language VM provides?
- An ISA VM enables code for one instruction set architecture to run on a machine w a different ISA
- An OS VM allows a guest operating system to run on a host operating system
- An HLL VM **allows execution of a high level language that converts to bytecode.**

Bytecode design decisions

- Fixed width: each instruction takes up the same number of bytes
- Variable width: instructions may consume a different number of bytes depending on the opcode.

What does a shell do?

- A program that exposes facilities of an operating system to the user.

Describe the UNIX "environment" and how it relates to BASH

- The environment in UNIX is a key/value structure that exists for every running process.
- The BASH variables are implemented in this environment.

What functionality of UNIX, made easy in BASH, allows you to compose programs?

- GO BACK TO THIS

Explain what a very basic BASH expression does

- #match upper case A and lower case b
  - grep "[Ab]" file.txt
- #match non-"A" characters
  - grep "[^aA]" file.txt
- #match one or more repeating ab's
  - grep "[ab]+" file.txt
- #match one or more repeating ab or bc's
  - grep "[ab|bc]+" file.txt