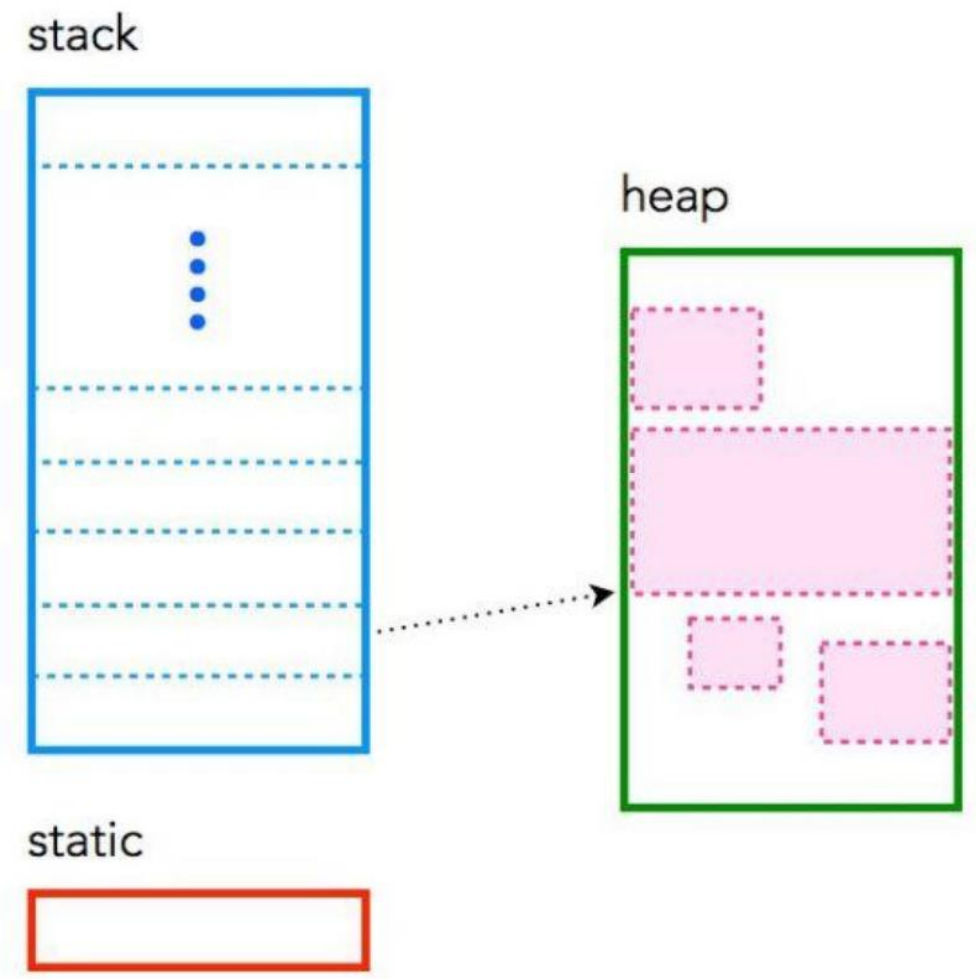
Q2 :



– static: global variable storage, permanent for the entire run of the program.

– stack: local variable storage (automatic, continuous memory).

– heap: dynamic storage (large pool of memory, not allocated in contiguous order).

A summary of the stack:

● the stack is managed by the CPU, there is no ability to modify it

● variables are allocated and freed automatically

● the stack it not limitless – most have an upper bound

● the stack grows and shrinks as variables are created and destroyed

● stack variables only exist whilst the function that created them exists

A summary of the heap:

● the heap is managed by the programmer, the ability to modify it is somewhat boundless

● in C, variables are allocated and freed using functions like malloc() and free()

● the heap is large, and is usually limited by the physical memory available

● the heap requires pointers to access it

What is stored in the static heap, stack, dynamic heap?

In Java, static variables and methods are stored in the static heap, which is a memory area associated with the class itself. It holds data that is shared among all instances of the class.

The stack is used for method execution and storing local variables and method call information.

The dynamic heap is the memory area where objects are allocated. It is managed by the Java runtime and is used to store objects and their instance variables.

What are objects in the program?

Objects are instances of classes that encapsulate data and behavior. In this program, objects are created for classes such as Vase, Statue, and Painting.

What is the item variable storing?

The item variable is of type Item, which is a superclass for Vase, Statue, and Painting. It can store objects of any of these classes or their subclasses. It is used to hold the current item being created or displayed.

Why must you cast to call the method inputVase()/outputVase()?

The item variable is declared as Item type, which is a superclass. However, the inputVase() and outputVase() methods are specific to the Vase class. To call these methods on the item object, we need to explicitly cast it to the Vase type, indicating that it is an instance of the Vase class or one of its subclasses.

What is the error thrown when you cast it wrong?

If you cast the item object to the wrong type, such as trying to cast it to a type that is not a superclass or a compatible subclass, a ClassCastException will be thrown at runtime.

What methods can you call if you don't cast the item variable?

If you don't cast the item variable, you can only call the methods defined in the Item class or its superclasses. The methods specific to Vase, Statue, or Painting will not be accessible without casting to the appropriate type.