Durante l’esecuzione di un programma in Java la memoria viene divisa in due parti:

* Stack
* Heap

Le due aree hanno scopi e caratteristiche differenti.

Infatti all’interno dello stack vengono memorizzare le variabili di tipo primitivo, gli indirizzi di memoria che puntano agli oggetti all’interno dell’heap. Ogni volta che viene chiamato un metodo si crea un nuovo record di attivazione. Dentro ad ogni record di attivazione vengono memorizzati i parametri, l’oggetto a cui appartiene (this), le variabili del metodo, il valore di ritorno, l’indirizzo della chiamata del metodo.   
I dati all’interno dello stack hanno uno scope di visibilità locale.  
Lo stack ha una dimensione limitata di pochi megabyte. Se viene superata si genera un errore di StackOverflowError.  
Lo stack si comporta come una pila, cioè, in modalità LIFO (Last Input First Output).   
  
  
  
Heap

L’heap è l’area di memoria in cui sono memorizzati gli oggetti e i loro attributi.  
Dato che gli oggetti sono memorizzati in modo sparso all’interno dell’heap, al momento della creazione di un nuovo oggetto, viene memorizzato nello stack l’indirizzo di memoria in cui si trova.

All’interno dell’heap sono memorizzati in modo non contiguo.

Infatti è il garbage collector che pensa a gestire lo spazio e che libera gli oggetti che non vengono più utilizzati.  
Gli elementi all’interno dell’heap sono accessibili in qualunque punto del programma.  
Se la memoria all’interno dell’heap si riempie java genera l’errore java.lang.OutOfMemoryError.  
L’heap è suddiviso in più parti:

* Young generation
* Survivor space
* Old generation
* Permanent generation
* Code Cache