Stack Vs Heap

1. $\boldsymbol{Like\ Stack, Heap\ is\ stored\ in\ RAM.}$

Suppose:

$$\boldsymbol{int * p = (int *) malloc\big(4 * sizeof(int)\big);}$$

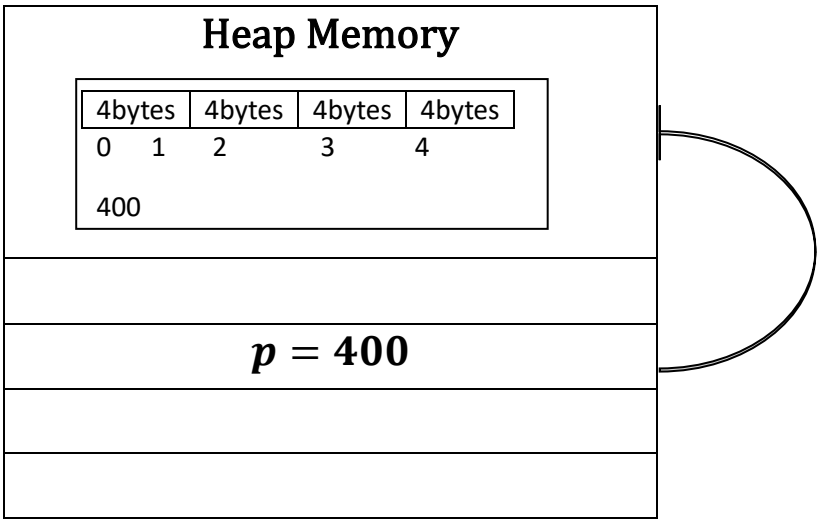
$\boldsymbol{It\ creates\ a\ space\ for\ an\ array\ of\ 4\ integers.}$

$\boldsymbol{But\ it\ get\ stored\ in\ Free\ Memory\ of\ RAM\ i.e.}$

$\boldsymbol{Heap\ Memory.}$

$\boldsymbol{Suppose\ *p\ holds\ memory\ address: 400, it\ get\ stored\ in\ Stack.}$

$\boldsymbol{The\ memory\ address\ 400\ is\ the\ address\ of\ Heap\ Memory\ where\ array\ of\ 4\ inetegers\ is\ stored.}$

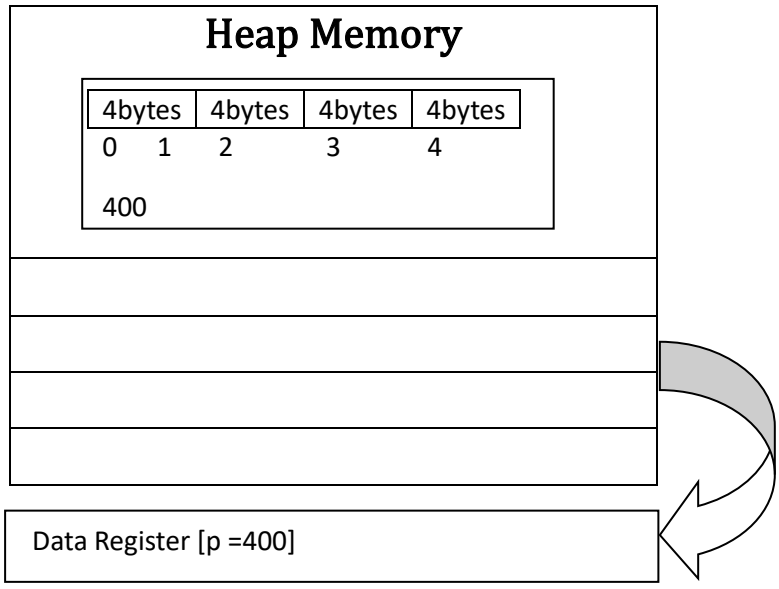


After operation in stack is finished:

- pointer p gets destroyed i.e., after Stack operation everything in stack gets popped out and stack gets empty.

- The data in heap memory remains at the location. Which we delete manually as discussed in arrays.

i.e.



i.e., free keyword for malloc and delete keyword for new operator.

Sometimes after deletion of data from stack we do is assign P to NULL after the operation gets over.

$$P = NULL\ .$$

To free the address or what so ever stored in pointer p variable.