Lecture : Memory Management (theory)

Agenda

- Basics of memory management
  - Stack
  - Call stack
  - Heap
- Problems | Quizzes

Expectation: Basics are clear.

- Recursion → call stacks
- Linked list | Trees | Tries | graph — Heap space

Introduction to stacks

LIFO : Last in first out.

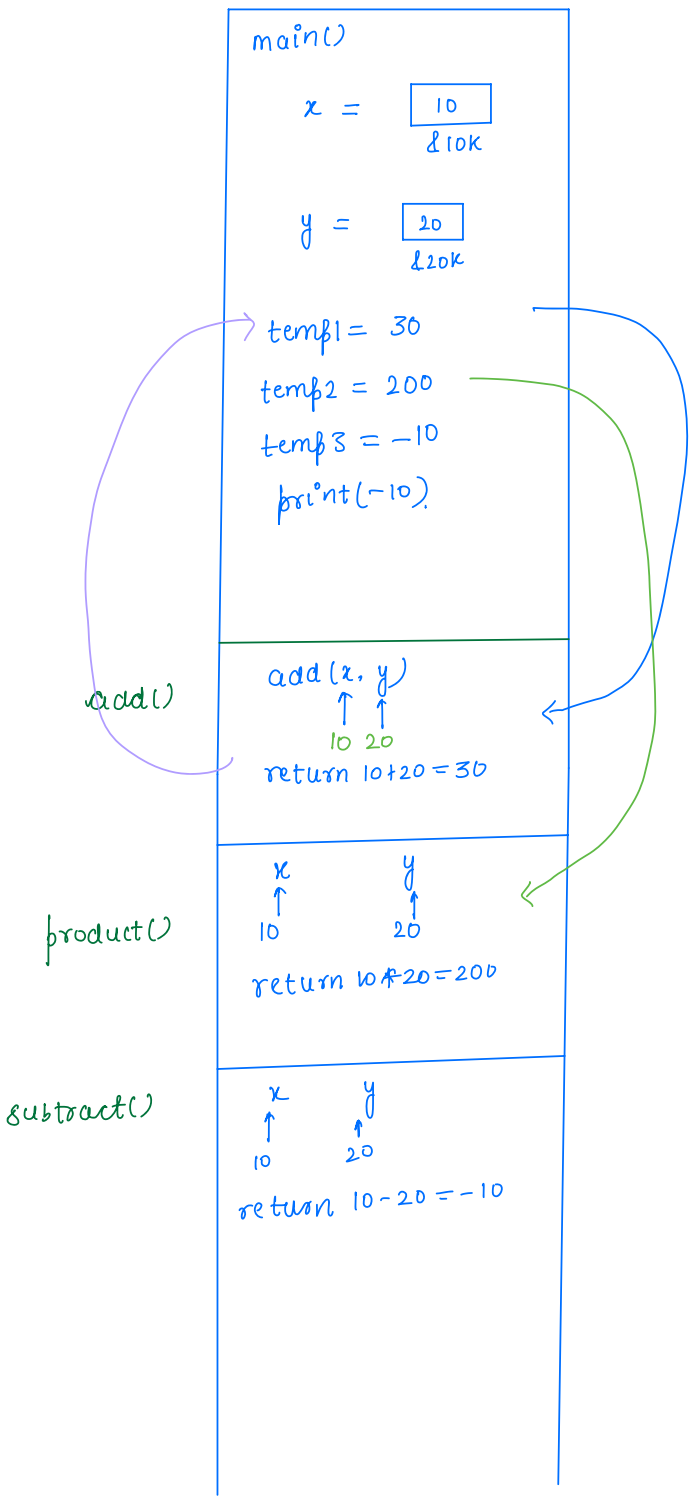
stack of plates.

" " books.

Introduction to call stack

→ Stores function.

```
int add(int x, int y) {
    return x+y;
}
int product(int x, int y) {
    return x*y;
}
int subtract(int x, int y) {
    return x-y;
}
Entry → main() {
    int x = 10;
    int y = 20;
    int temp1 = add(x, y);
    int temp2 = product(x, y);
    int temp3 = subtract(x, y);
    print(temp3);
}
```



```
int add(int x, int y) {
    return x+y;
}
main() {
    int x=10;
    int y = 20;
    int temp1 = add(x, y);
    int temp2 = add(temp1, 30);
    int temp3 = add(temp2, 40);
    print(temp3);
}
```

main()

x = 10
& 10K

y = 20
& 20K

temp1 = 30

temp2 = 60

temp3 = 100

print(100)

add()

x	y
10	20

return 10+20=30

add()

x	y
30	30

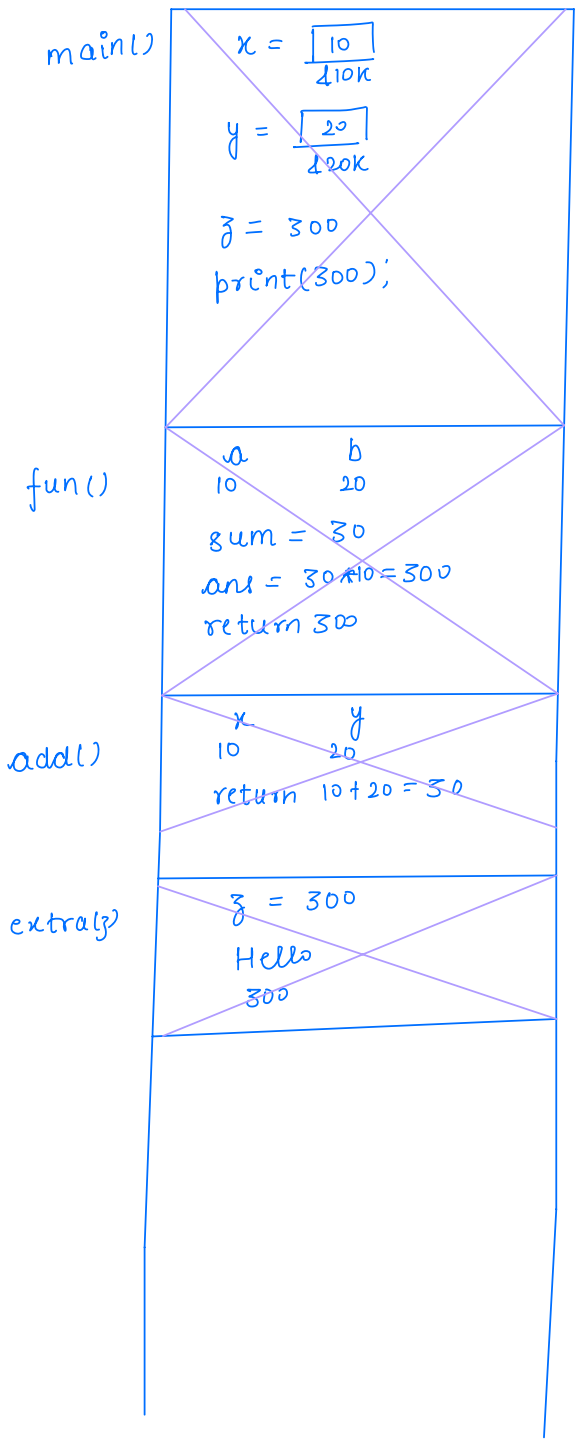
return 30 + 30 = 60

add()

x	y
60	40

return 60 + 40 = 100

```
int add(int x, int y) {
    return x+y;
}

int fun(int a, int b) {
    int sum = add(a, b);
    int ans = sum*10;
    return ans;
}

void extra(int w) {
    print("Hello");
    print(w);
}

main() {
    int x=10;
    int y = 20;
    int z = fun(x,y);
    print(z);
    extra(z);
}
```



Types of memory in Java

- Stack ⟹ primitive data types [int, char, float]
  - reference/address.
- Heap ⟹ Objects {anything initialised by new keyword
  - Arrays
  - ArrayList
  - Student
  - Hashmap, Hashset etc.

```
void main() {
    int x = 10;
(Heap) int[] arr = new int[3];
    print(arr);
    print(arr[2]); ⟶ &808
    arr[1] = 7;
}
```

call stack

main()

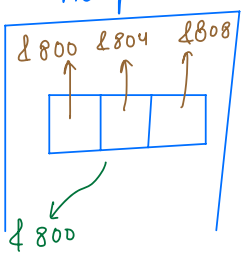
x = 10 (&1000)

arr = &800

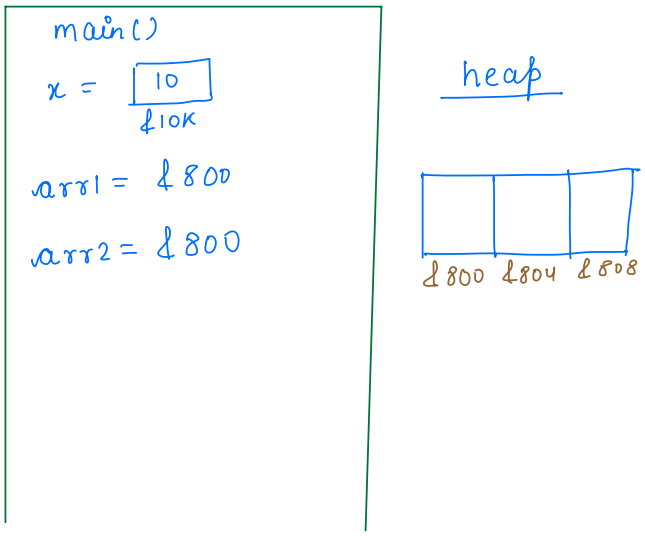
print(&800)

print(0)

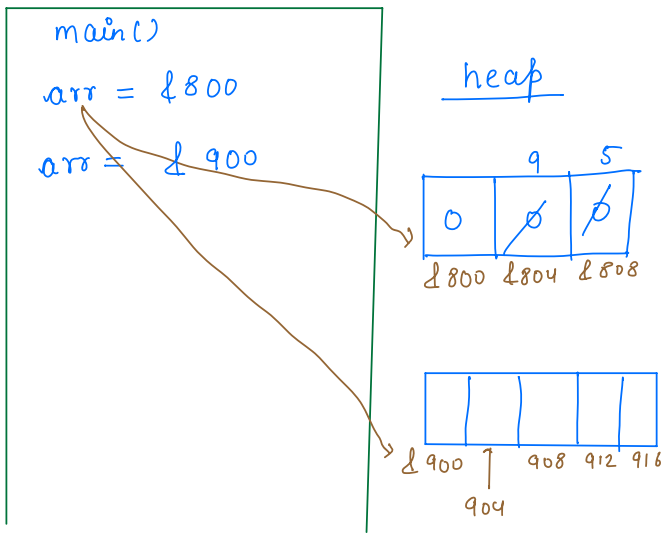
heap



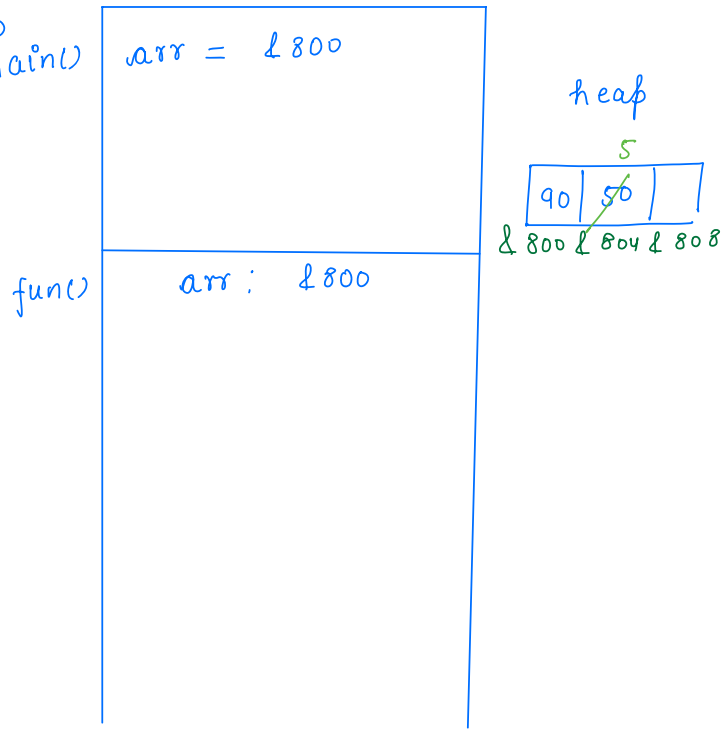
```
main() {
    int x = 10;
    int[] arr1 = new int[3];
    int[] arr2 = arr1;
    print(arr1); // &800
    print(arr2); // &800
}
```




```
main() {
    int[] arr = new int[3];
    print(arr); // &800
    arr[1] = 9;
    arr[2] = 5;
    arr = new int[5];
    print(arr[1]); // 0
    print(arr[2]); // 0
    print(arr); //&900
}
```



```
void fun(int[] arr) {
    print(arr); // &800
    arr[1] = 5;
}

main() {
    int[] arr = new int[3];
    print(arr); // &800
    arr[0] = 90;
    arr[1] = 50;
    fun(arr);
    print(arr[1]); // 5
}
```



```
main() {
    float y = 7.84f;
    int[][] mat = new int[3][4];
    print(mat);
    print(mat[1]);
    print(mat[1][3]);
}
```

3 arrays of size 4.

`print(mat);` || &2000

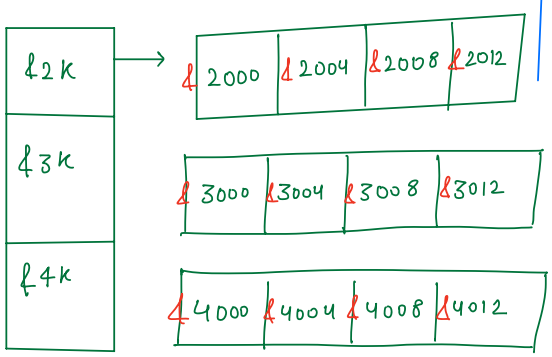
`print(mat[1]);` || &3000

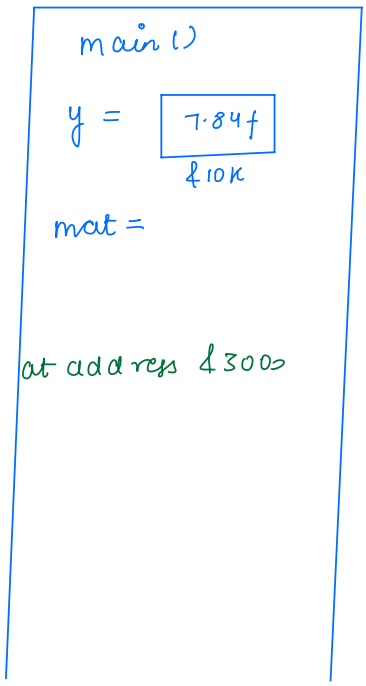
array[4] starting at address &3000

`print(mat[1][3]);` || 0

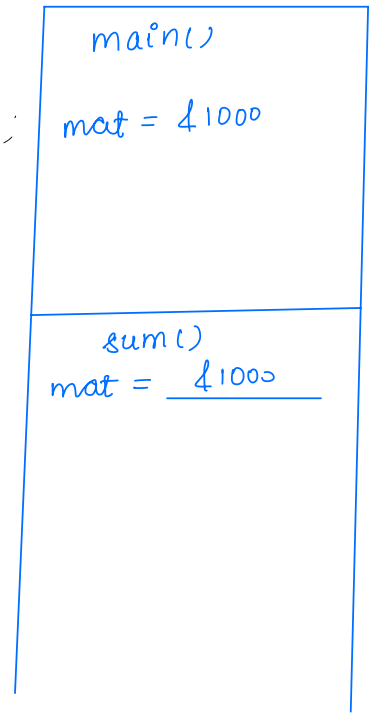
arr[3] ⟹ value.

heap

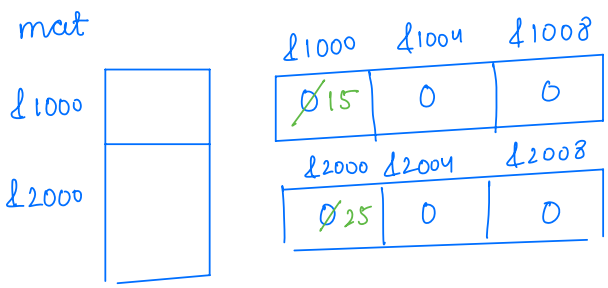




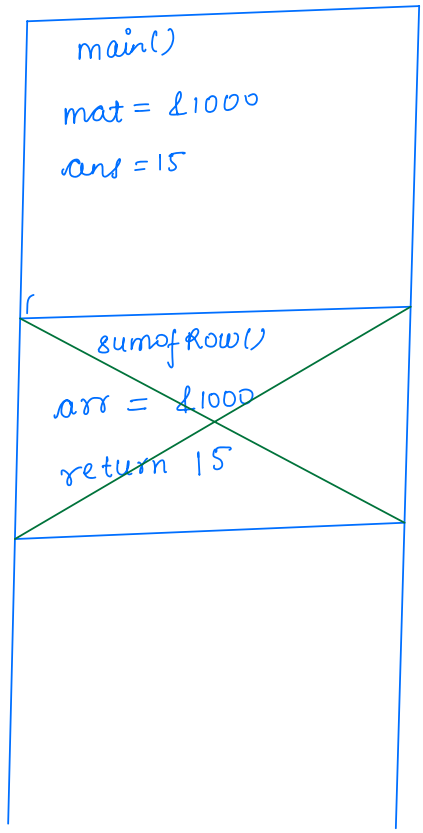
```
void sum(int[][] mat) {
    print(mat);                    // &1000
    print(mat[0][0] + mat[1][0]);
                                   // 40
}

main() {
    int[][]mat = new int[2][3];
    mat[0][0] = 15;
    mat[1][0] = 25;
    sum(mat);
}
```

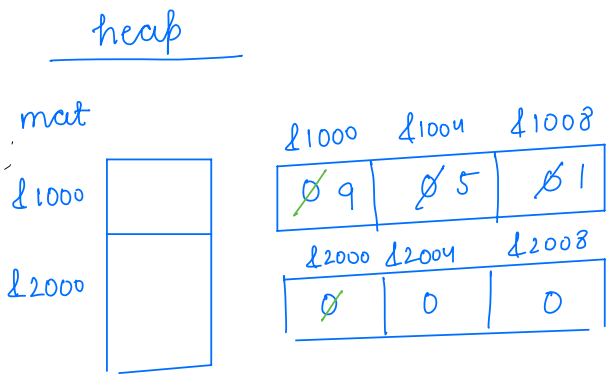


heap



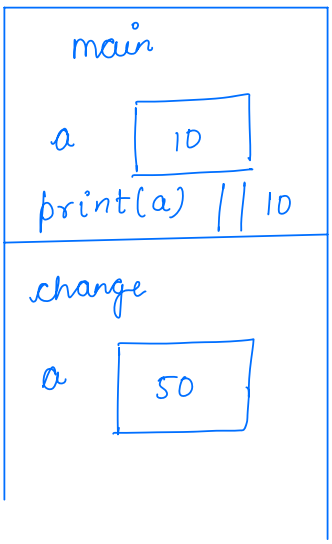
```
int sumOfRow( int[] arr ) {
    print(arr);  // &1000
    int sum=0;
    for(i=0 ; i< arr.length; i++) {
        sum = sum + arr[i];
    }
    return sum;
}

void main() {
    int[][] mat = new int[2][3];
    mat[0][0] = 9;
    mat[0][1] = 5;
    mat[0][2] = 1;
    int ans = sumOfRow(mat[0]);
    print(ans);  // 15
}
```





Quiz 1

```
void change(int a) {

    a = 50;
    print(a) || 50
}

main() {
    int a = 10;

    change(a);

    print(a); || 10
}
```



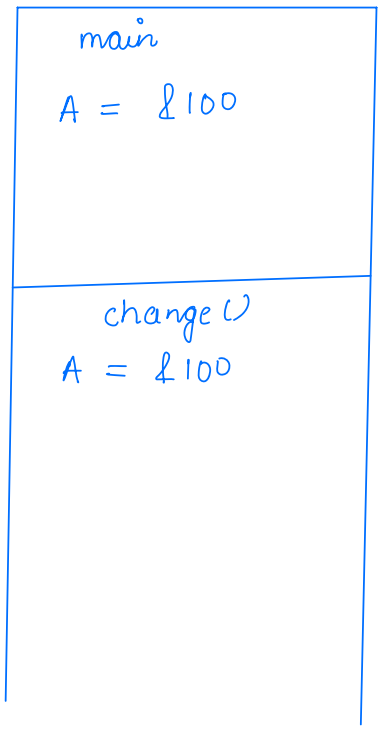
Quiz2

```
void change(int[] A){
    A[0] = 50;
}

main() {
    int[] A = {10};
    change(A);
    print(A[0]); // 50
}
```

heap

A $\boxed{\cancel{10}\ 50}$

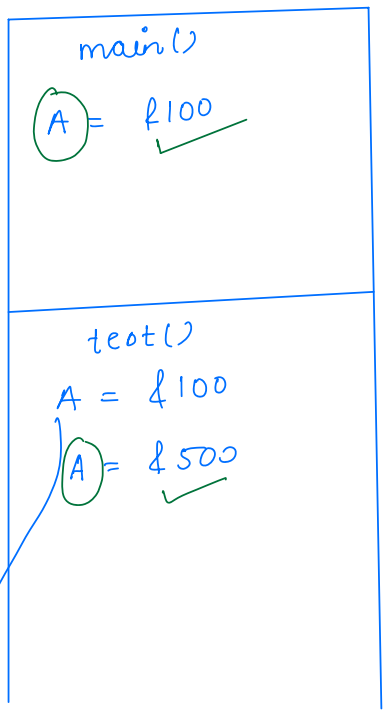
&100

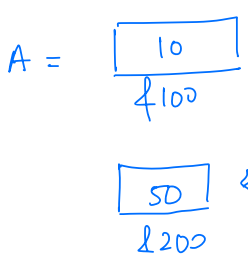


print(a); //

Quiz3

```
void test(int[] A) {
    A = new int[1];
    A[0] = 50;
}

main() {
    int[] A = {10};
    test(A);
    print(A[0]); //10
}
```





Quiz 4

```
void fun(int[] A) {
    A = new int[1];
    A[0] = 100;
}

main() {
    int[] A = { 10, 20, 30 };
    fun(A);
    print(A[0]);   // 10
}
```

main()
A = &100

fun()
A = &100
A = &200

heap

A = [10 | 20 | 30]
&100 &104 &108

A = [100]
&200

Quiz 5

```
void swap( int a, int b){
    int temp = a;
    a = b;
    b = temp;
}
main() {
    int a = 10;
    int b = 20;
    swap( a, b);
    print( a + " " + b);
}
```

h | w

swap

f

Quiz6

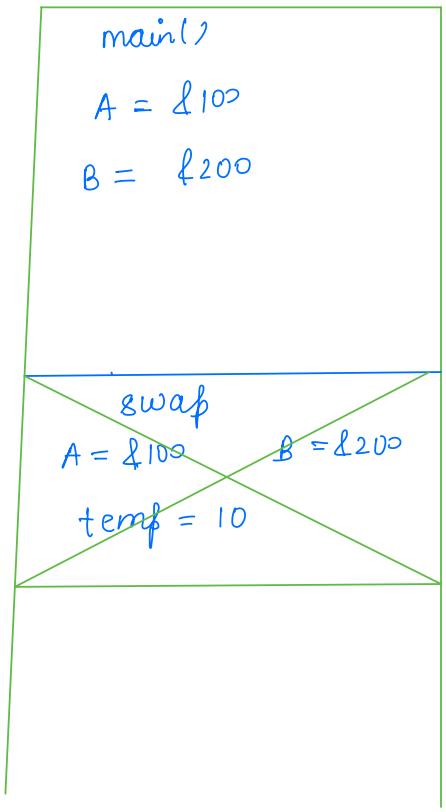
```
void swap ( int[] A , int[] B ) {
        int temp = A[0];
        A[0] = B[0];
        B[0] = temp;
}

main() {
    int[] A = {10};
    int[] B = {20};
    swap( A , B);
    print( A[0] + " " + B[0]);
}
```

20 10

heap

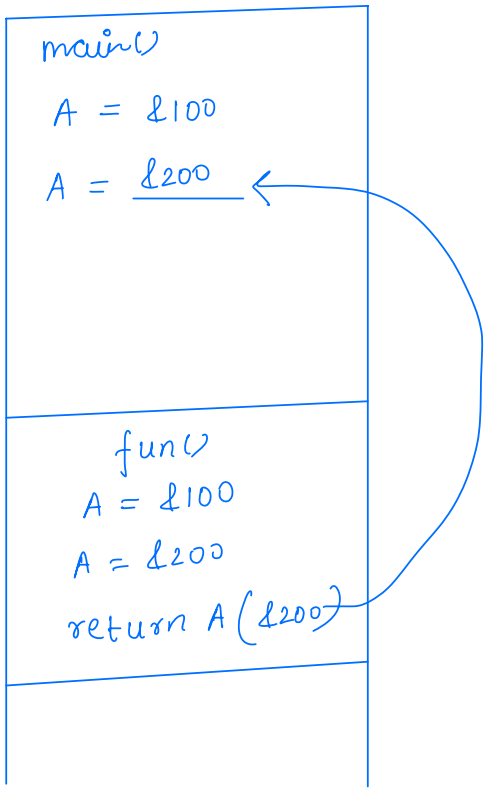
A = [~~10~~ 20] &100

B = [~~20~~ 10] &200

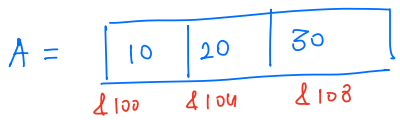


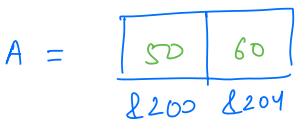
Quiz 7

```
int[] fun(int[] A) {
    A = new int[2];
    A[0] = 50;
    A[1] = 60;
    return A;
}

main() {
    int[] A = {10, 20, 30};
    A = fun(A);
    print(A[0]); // 5
}
```



heap





Quiz8

```
void test(int[] A) {
    A = new int[2];
    A[0] = 94;
}

main() {
    int[] A = {10, 20, 30};
    test(A);
    print(A[0]); // 10
}
```

heap

A = | 10 | 20 | 30 |

&100 &104 &108

A = | 94 | |

&200 &204

main()

A = &100

test(A)

&100

A = &200

Thank you ☺

Doubt

String[] arr = { Aman, Ayush, raghu }