

Let us move ahead by considering the remaining cases....

Case-2: With input and without output

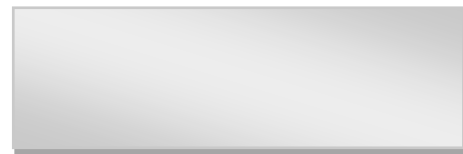
Code Segment

```
class Addition
{
    int c;

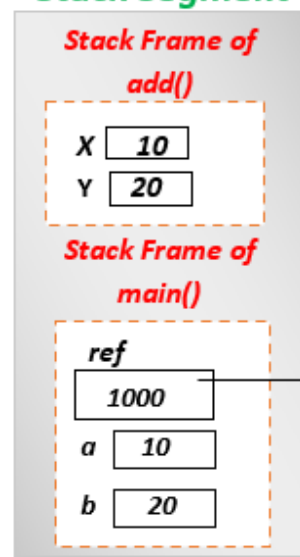
    void add(int x,int y)
    {
        c = x + y;
        System.out.println(c);
    }
}

class Demo
{
    public static void main(String[] args)
    {
        Addition ref = new Addition();
        int a,b;
        a = 10;
        b = 20;
        ref.add(a,b);
    }
}
```

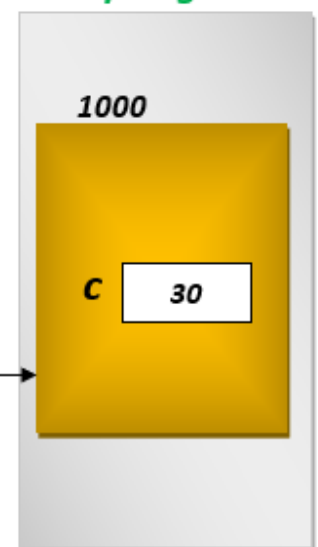
Static Segment



Stack Segment



Heap Segment



Output:

30

Explanation:

Here, the execution starts from main method() which is called by the Operating System. Whenever, a method is called a region is created in the stack segment called **Stack frame**. And therefore, stack frame of **main()** gets created on the stack segment.

By using a **"new"** keyword an object is created and memory for it is allocated in the heap segment. The instance variable **c** is allocated memory in the heap segment, local variable **x,y,a,b** are allocated memory on stack segment on their respective stack frames and default values are given to instance variables by the JVM. A reference variable is created with name **ref** and is created in Stack segment. Now, **add()** is a method which accepts two parameters as inputs. This method **add()** is called and stack frame of **add()** is created in the Stack segment. Then, the body of **add()** is executed, now this doesn't return any value hence, control goes back to the caller of the method.

Case-3: Without input and with output

Code Segment

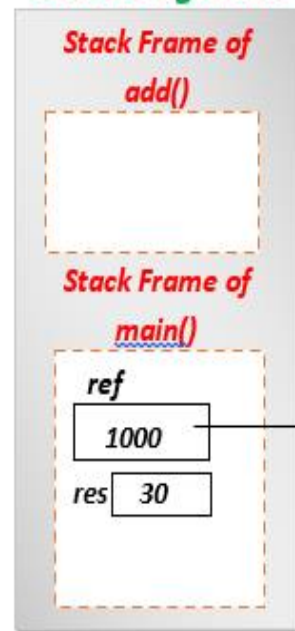
```
class Addition
{
    int a,b,c;

    int add()
    {
        a = 10;
        b = 20;
        c = a + b;
        return c;
    }
}

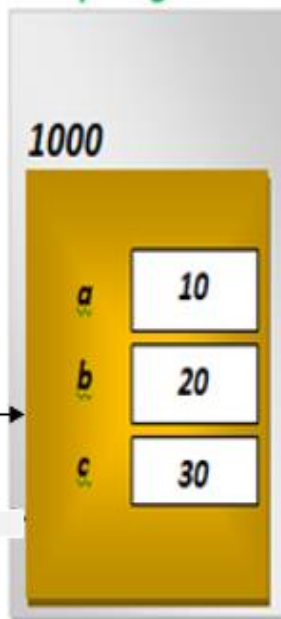
class Demo
{
    public static void main(String[] args)
    {
        Addition ref = new Addition();
        int res;
        res = ref.add();
        System.out.println(res);
    }
}
```

Static Segment

Stack Segment



Heap Segment



Output:

30

Explanation:

Here, the execution starts from main method() which is called by the Operating System. Whenever, a method is called a region is created in the stack segment called **Stack frame**. And therefore, stack frame of **main()** gets created on the stack segment.

By using a **"new"** keyword an object is created and memory for it is allocated in the heap segment. The instance variable **a,b,c** are allocated memory in the heap segment, local variable **res** allocated is memory on stack segment in stack frame of **main()** and default values are given to instance variables by the JVM. A reference variable is created with name **ref** and is created in Stack segment.

Now, **add()** is a method which accepts 0 parameters as inputs. This method **add()** is called and stack frame of **add()** is created in the Stack segment. Then, the body of **add()** is executed, now this method return **c** value which is collected in **res** in **main()** and control goes back to the caller of the method.

Case-4: With input and with output

Code Segment

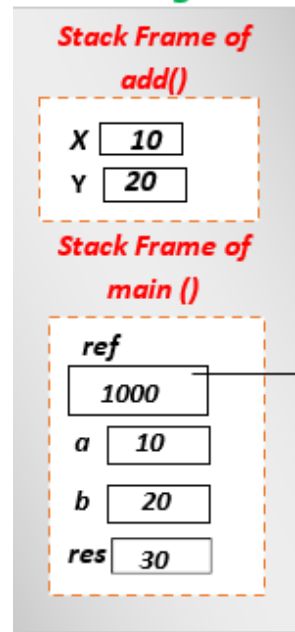
```
class Addition
{
    int c;

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

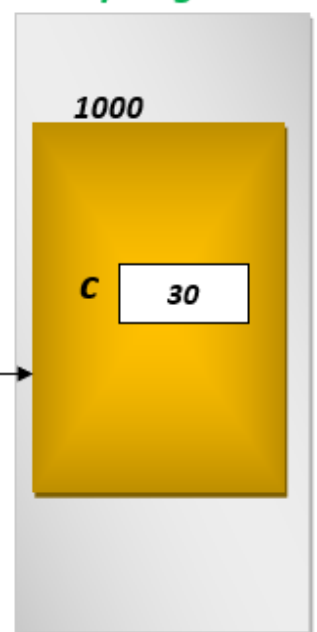
class Demo
{
    public static void main(String[] args)
    {
        Addition ref = new Addition();
        int a,b,res;
        a = 10;
        b = 20;
        res = ref.add(a,b);
        System.out.println(res);
    }
}
```

Static Segment

Stack Segment



Heap Segment



Output:

30

Explanation:

Here, the execution starts from main method() which is called by the Operating System. Whenever, a method is called a region is created in the stack segment called **Stack frame**. And therefore, stack frame of **main()** gets created on the stack segment.

By using a "**new**" keyword an object is created and memory for it is allocated in the heap segment. The instance variable **c** is allocated memory in the heap segment, local variable **a,b,x,y,res** allocated is memory on stack segment in stack frame and default values are given to instance variables by the JVM. A reference variable is created with name **ref** and is created in Stack segment.

Now, **add()** is a method which accepts 2 parameters as inputs. This method **add()** is called and stack frame of **add()** is created in the Stack segment. Then, the body of **add()** is executed, now this method return **c** value which is collected in **res** in **main()** and control goes back to the caller of the method.