**Question:**

Given the **root** of a Binary Search Tree and an integer **k**, write a function **smallest\_sum(root, k)** that returns the **sum of the k smallest nodes** of the BST.

You can assume **k** is **always valid** (0 ≤ k ≤ total number of nodes).

**Constraints:**

* You must solve it using only recursion.
* No extra data structures like arrays, lists, or stacks.

| **Example of BST** | **Sample Inputs** | **Sample Outputs** | **Explanation** |
| --- | --- | --- | --- |
|  | **smallest\_sum(root, 3)** | **8** | Sum 3 smallest nodes are: 1+3+4 = 8 |
| **smallest\_sum(root, 5)** | **21** | Sum 5 smallest nodes are: 1+3+4+6+7 = 21 |
| **smallest\_sum(root, 6)** | **29** | Sum 6 smallest nodes are: 1+3+4+6+7+8 = 29 |

**Question:**

Given the **root** of a Binary Search Tree and an integer **k**, write a function **smallest\_mul(root, k)** that returns the **multiplication of the k smallest nodes** of the BST.

You can assume **k** is **always valid** (0 ≤ k ≤ total number of nodes).

**Constraints:**

* You must solve it using only recursion.
* No extra data structures like arrays, lists, or stacks.

| **Example of BST** | **Sample Inputs** | **Sample Outputs** | **Explanation** |
| --- | --- | --- | --- |
|  | **smallest\_mul(root, 3)** | **12** | Sum 3 smallest nodes are: 1\*3\*4 = 12 |
| **smallest\_mul(root, 4)** | **72** | Sum 4 smallest nodes are: 1\*3\*4\*6 = 72 |
| **smallest\_mul(root, 6)** | **4032** | Sum 6 smallest nodes are: 1\*3\*4\*6\*7\*8 = 4032 |