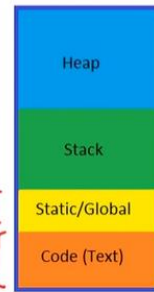


Pointers and dynamic memory

```
#include<stdio.h>
int total;
int Square(int x)
{
    return x*x;
}
int SquareOfSum(int x,int y)
{
    int z = Square(x+y);
    return z;
}
int main()
{
    int a = 4, b = 8;
    total = SquareOfSum(a,b);
    printf("output = %d",total);
}
```

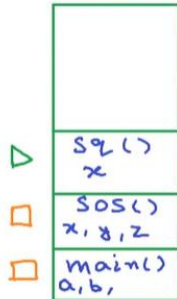
Application's memory



function calls
local variables
global
Instructions

```
#include<stdio.h>
int total;
int Square(int x)
{
    return x*x; // z^2
}
int SquareOfSum(int x,int y)
{
    int z = Square(x+y);
    return z; // (z+y)^2
}
int main()
{
    int a = 4, b = 8;
    total = SquareOfSum(a,b);
    printf("output = %d",total);
}
```

Stack



Stack-frame

Global



Application's memory



```
#include<stdio.h>
int total;
int Square(int x)
{
    return x*x; // z^2
}
int SquareOfSum(int x,int y)
{
    int z = Square(x+y);
    return z; // (z+y)^2
}
int main()
{
    int a = 4, b = 8;
    total = SquareOfSum(a,b);
    printf("output = %d",total);
}
```

Stack (1 MB)



Stackoverflow

Global



Application's memory



```
#include<stdio.h>
#include<stdlib.h>
int main()
{
    int a; // goes on stack
    int *p;
    p = (int*)malloc(sizeof(int));
    *p = 10;
    p = (int*)malloc(sizeof(int));
    *p = 20;
}
```

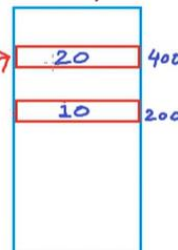
Stack



Global



Heap



Application's memory



Free Store