


Intro a C

Funciones

With  by @vichoeq & @KnowYourselfs

Funciones - C v/s Python



```
out_type name(arg1_type arg1, ..., argn_type argn)
{
    // Código
    return ...;
}
```

```
void name(...)
{
    // Función sin retorno
}
```



```
def name(arg1, ..., argn):
    # Código
```

Declarar v/s Definir



```
// Declarar
```

```
int foo(int bar);
```

```
int foo(int);
```

Declarar: Indicar el nombre de la función y sus *tipos*.

```
// Definir
```

```
int foo(int bar)
```

```
{
```

```
    return bar * 2;
```

```
}
```

Definir: Explicita el contenido de una función.

STACK de Memoria

STACK de memoria

Todo programa se encuentra en la memoria del computador. Dentro de este espacio, hay un sector especial llamado **STACK** que guarda lo siguiente:

- Llamados a bloques de código
- Variables de bloques de código

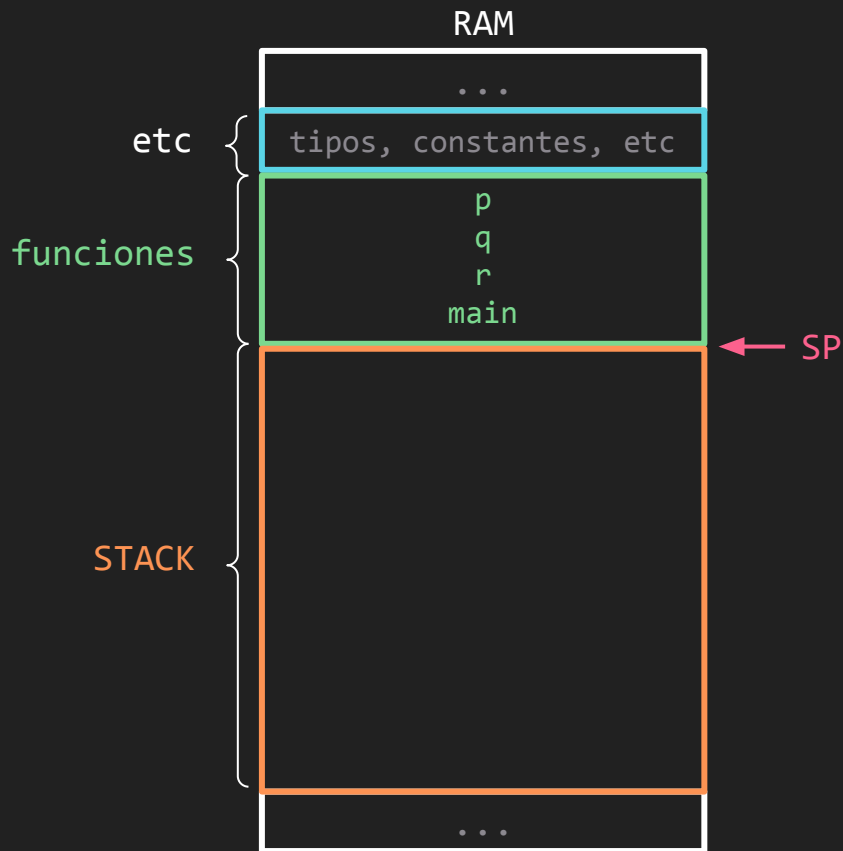
STACK de Memoria

Ejemplo #1

STACK de memoria



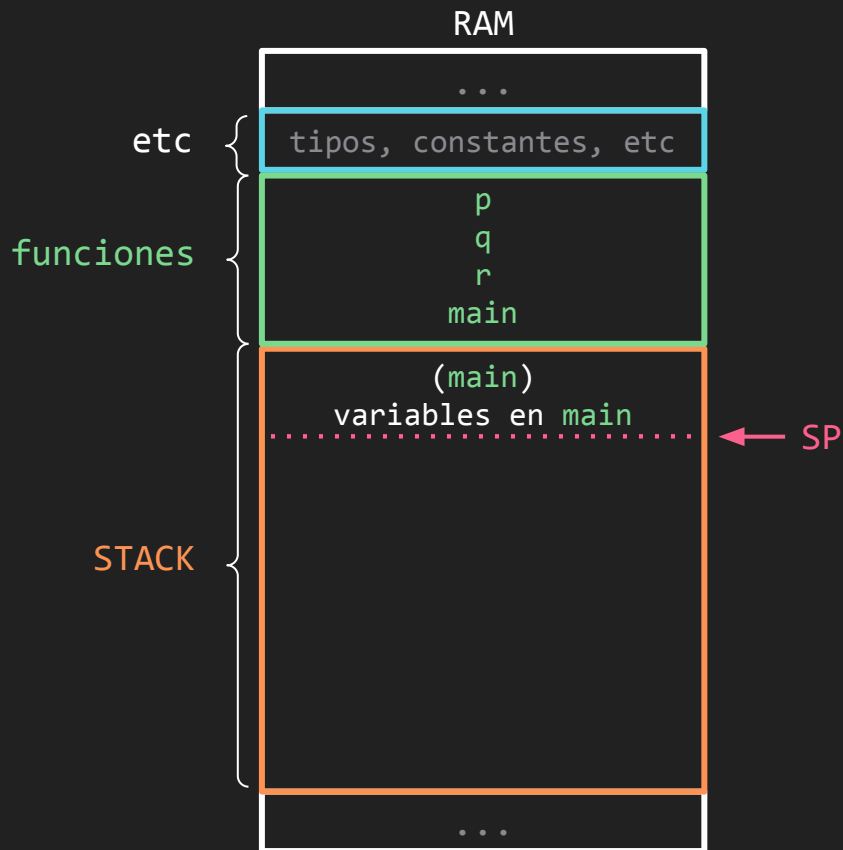
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



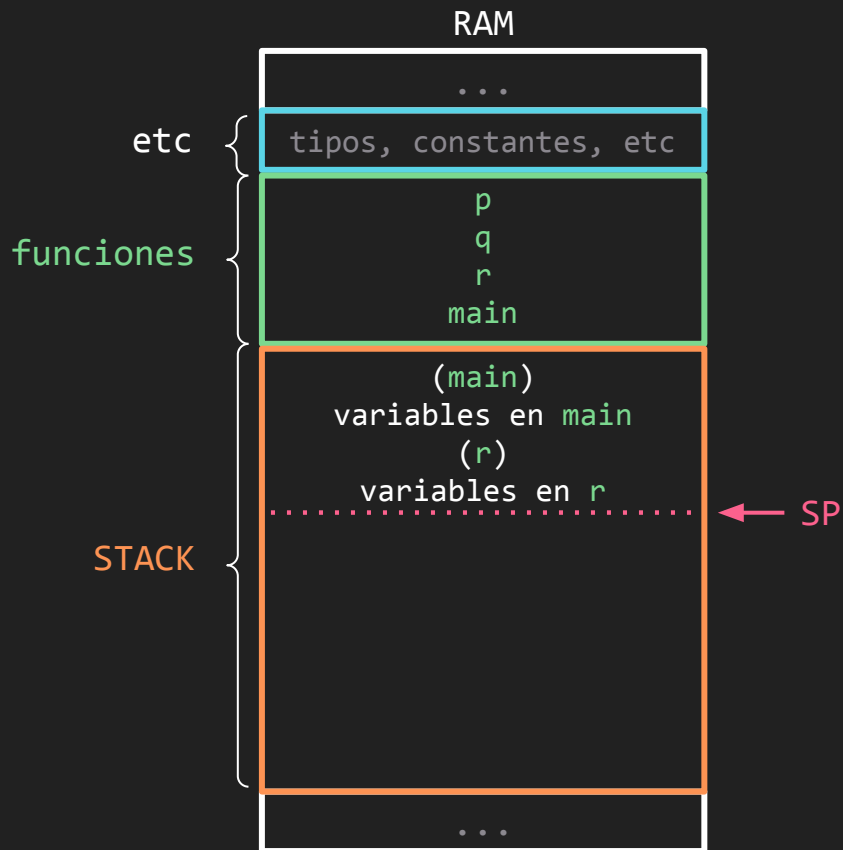
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



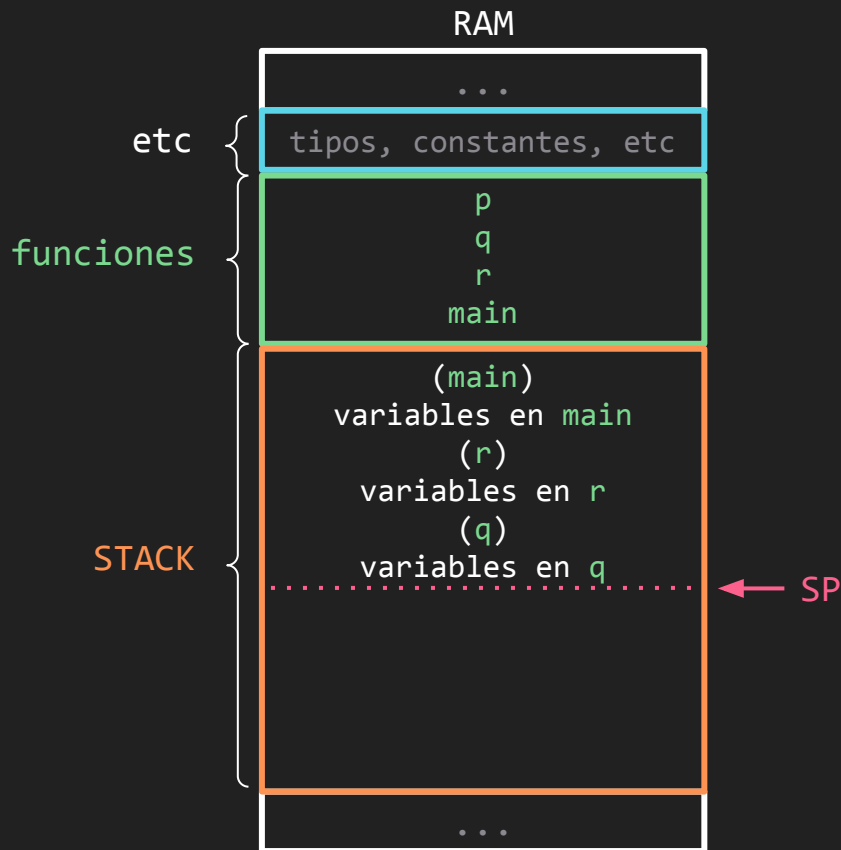
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



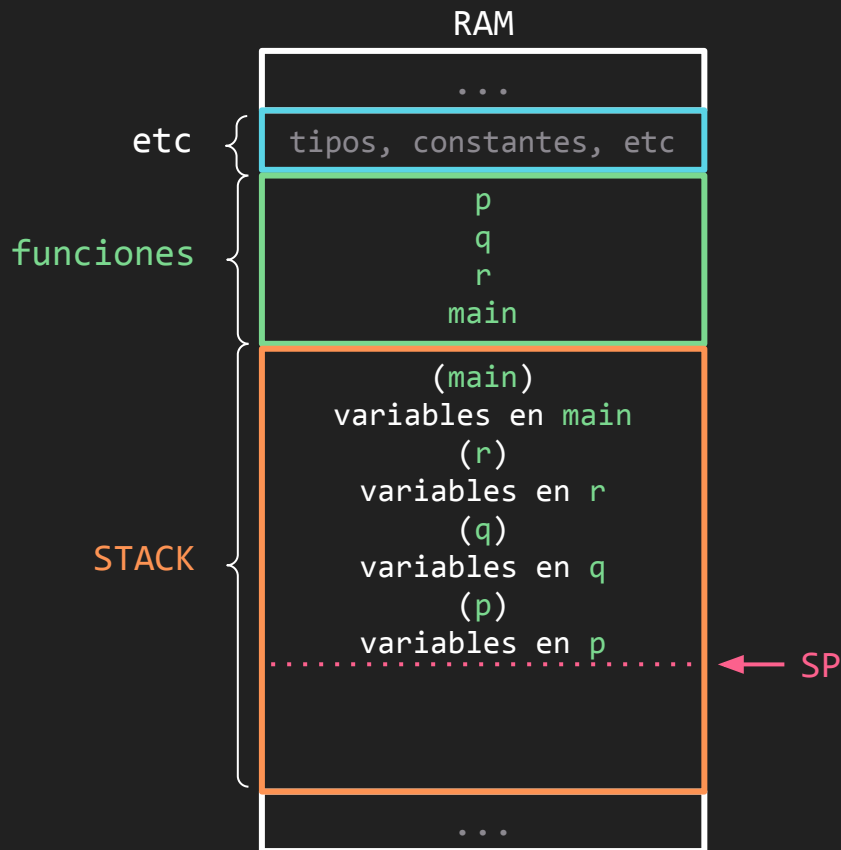
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



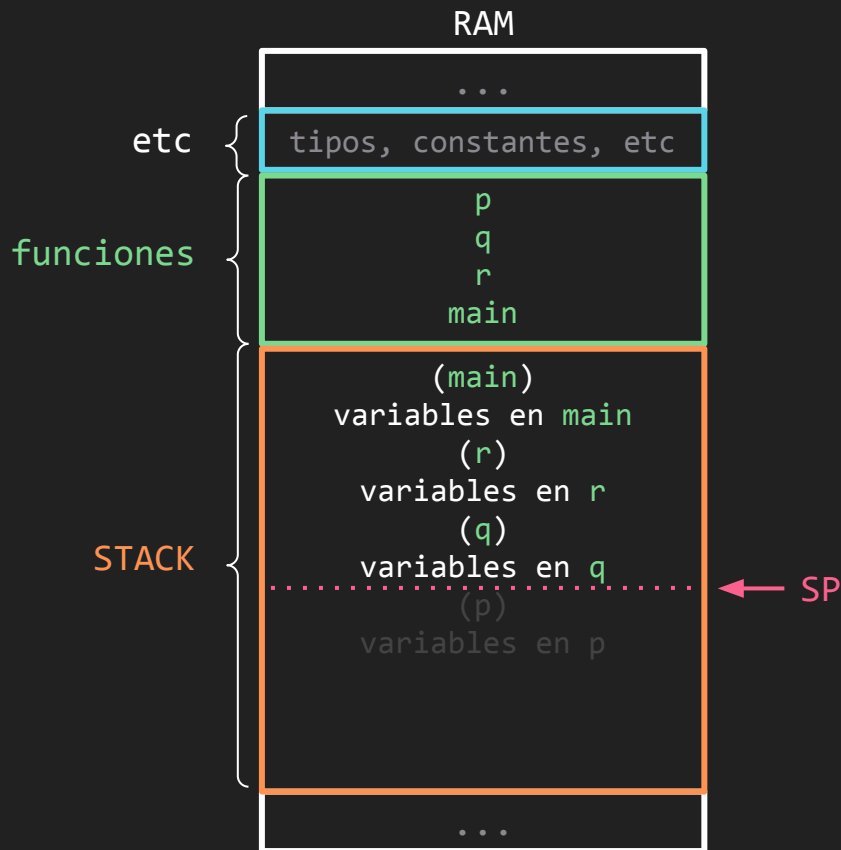
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



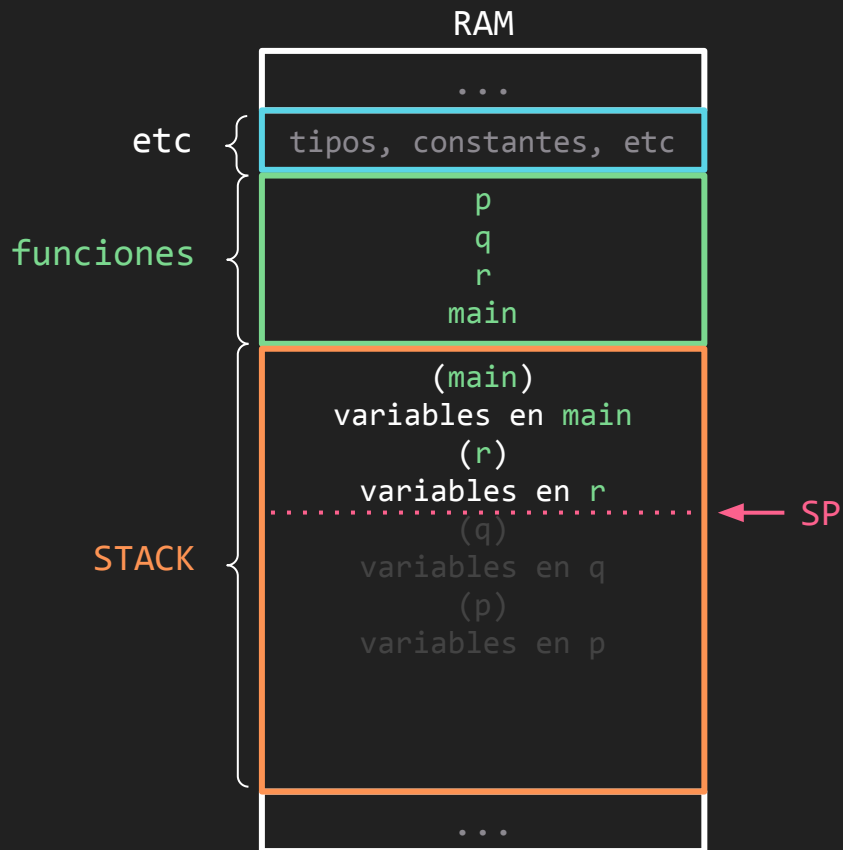
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



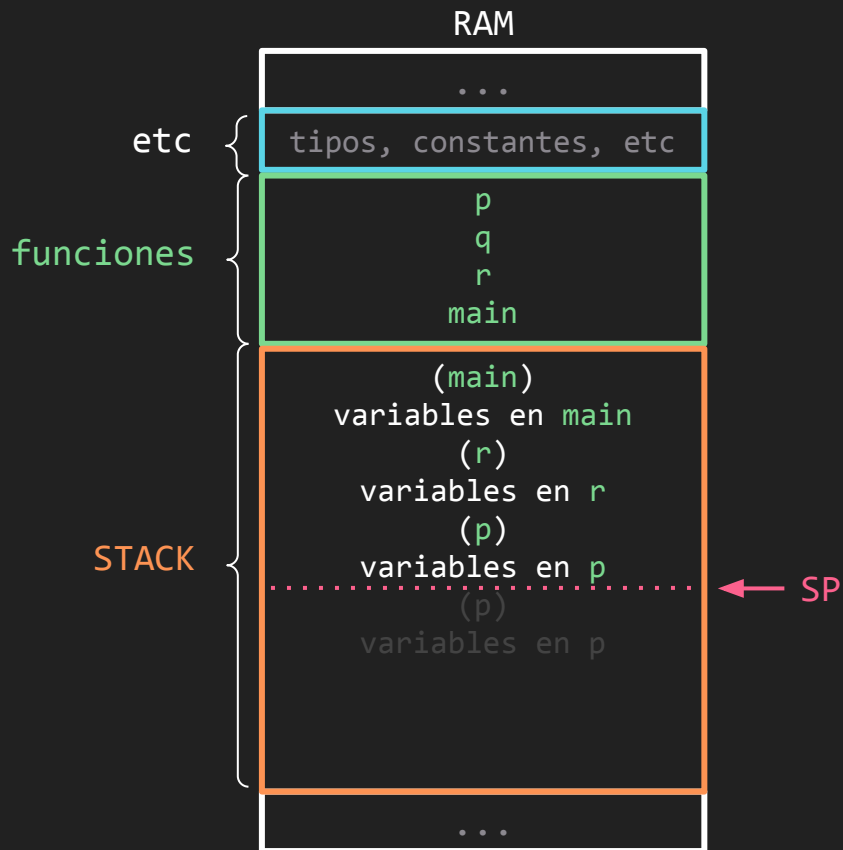
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



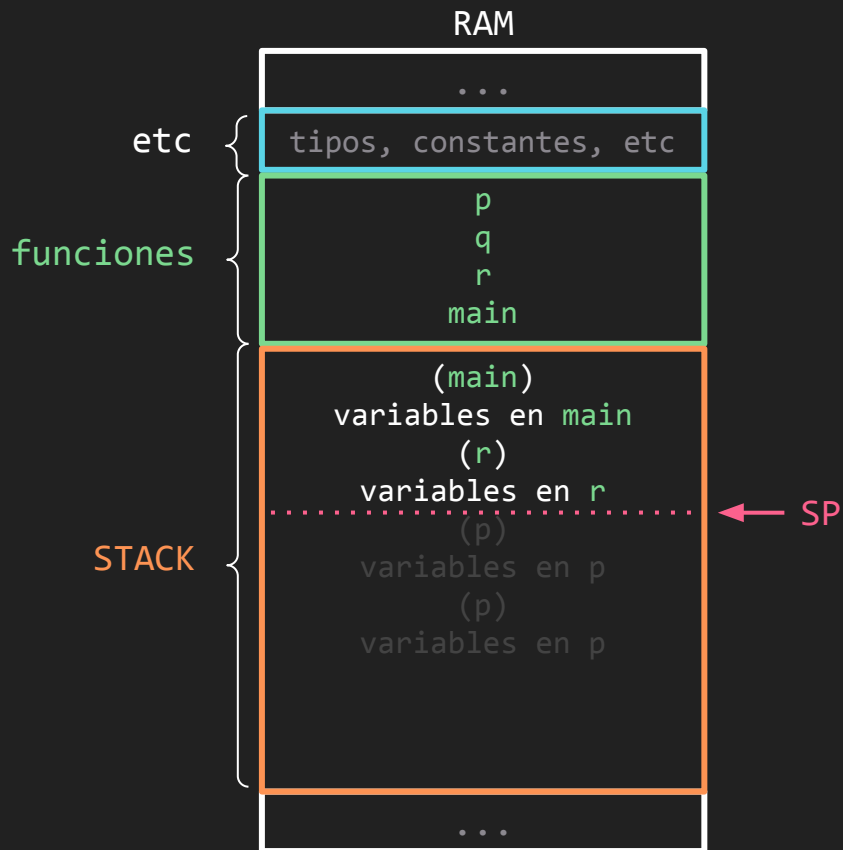
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



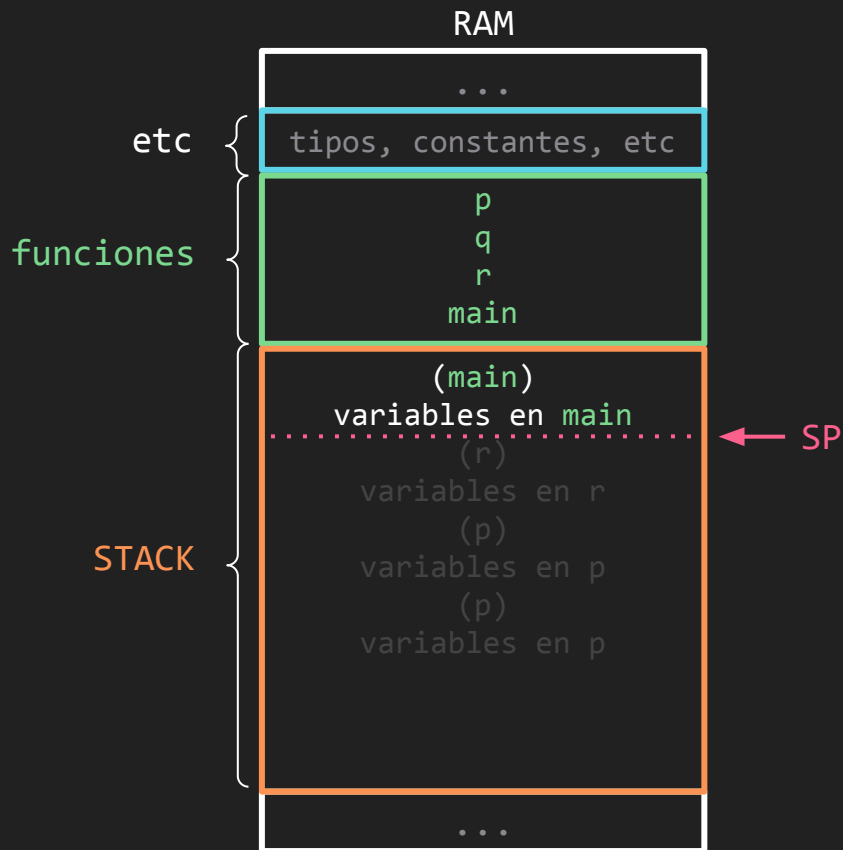
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



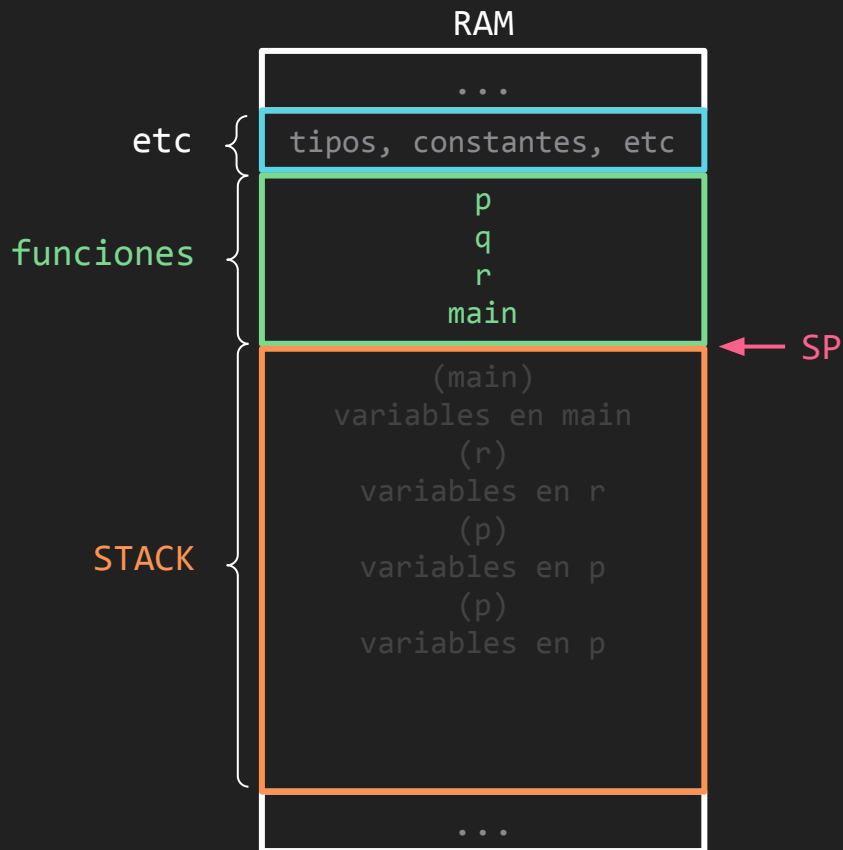
```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de memoria



```
int p(...);  
  
// Llama a p  
int q(...);  
  
// Llama a q y a p  
int r(...);  
  
// Llama a r  
int main();
```



STACK de Memoria

Ejemplo #2

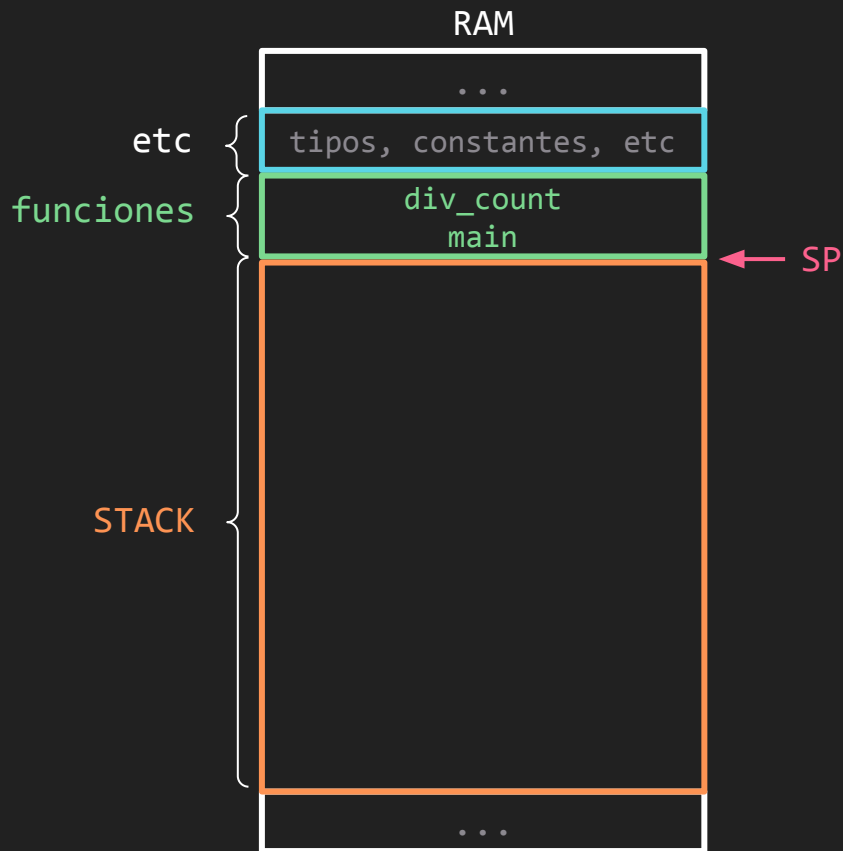
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



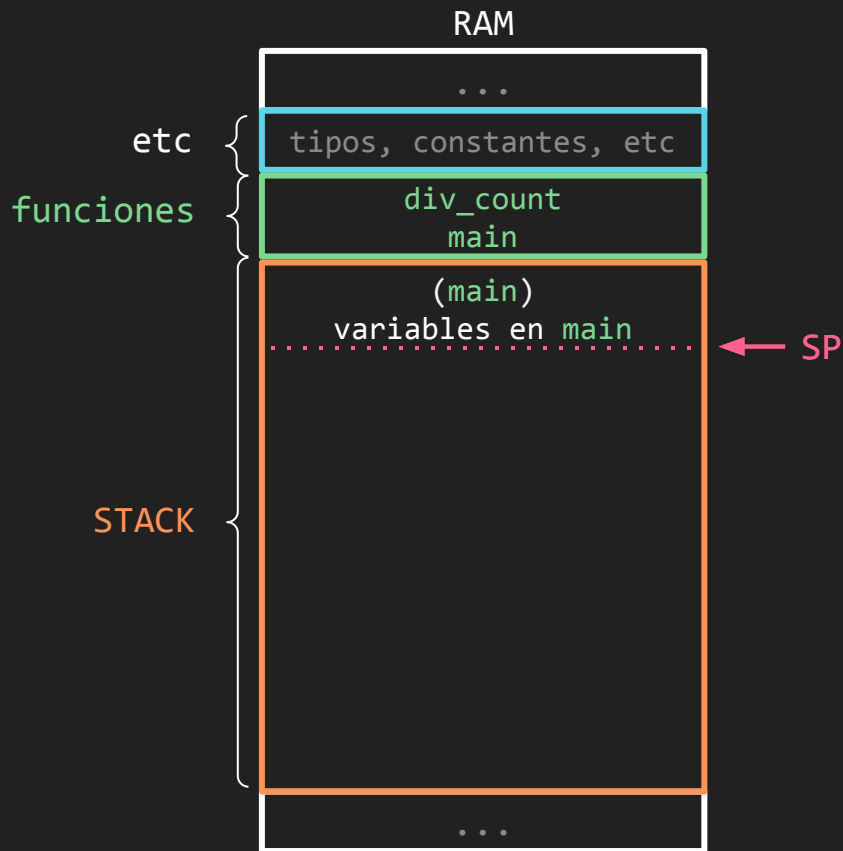
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



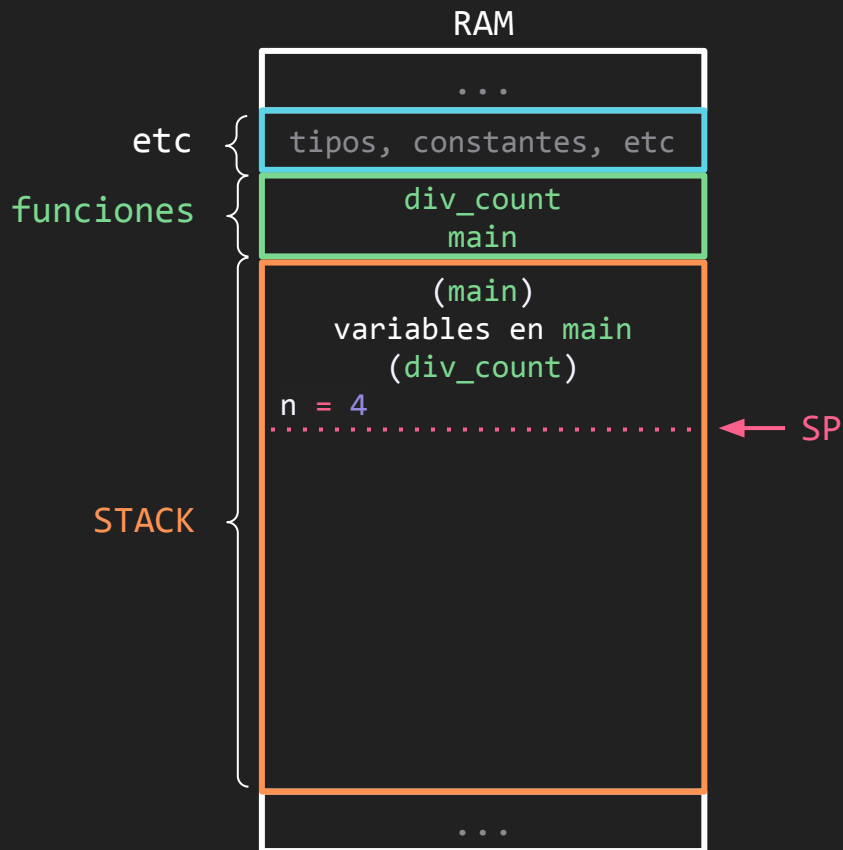
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



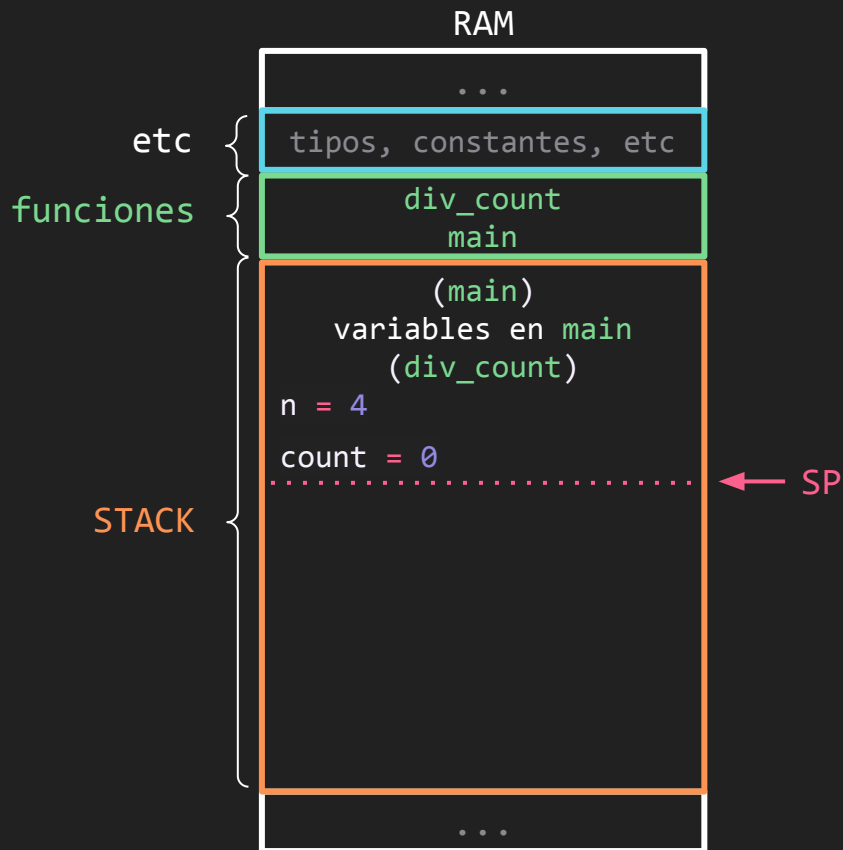
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



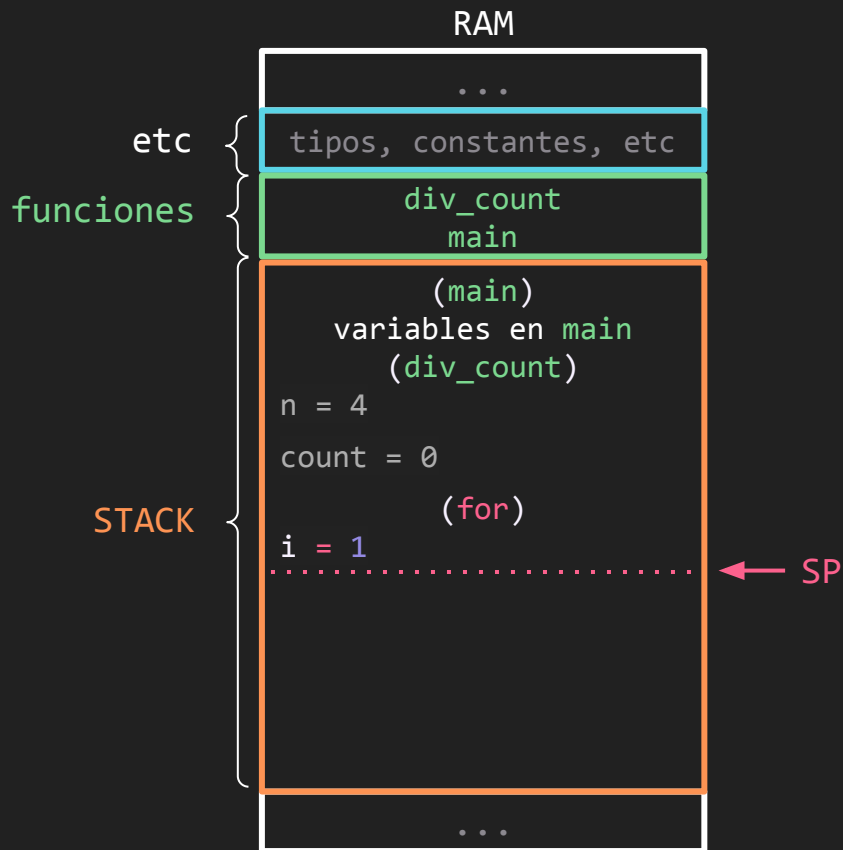
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



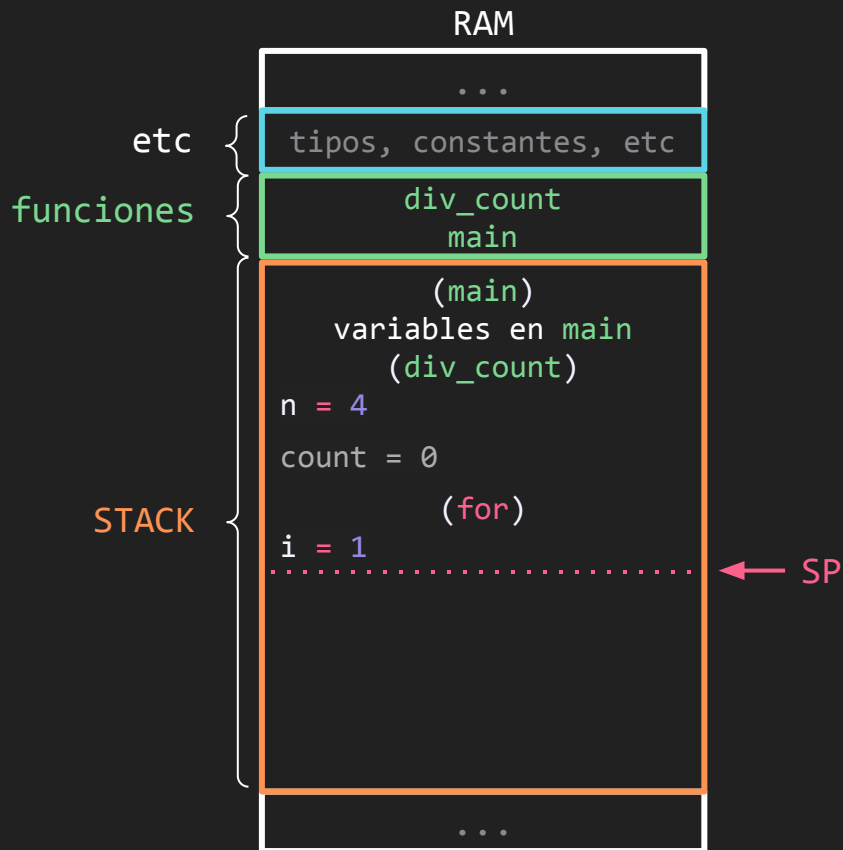
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



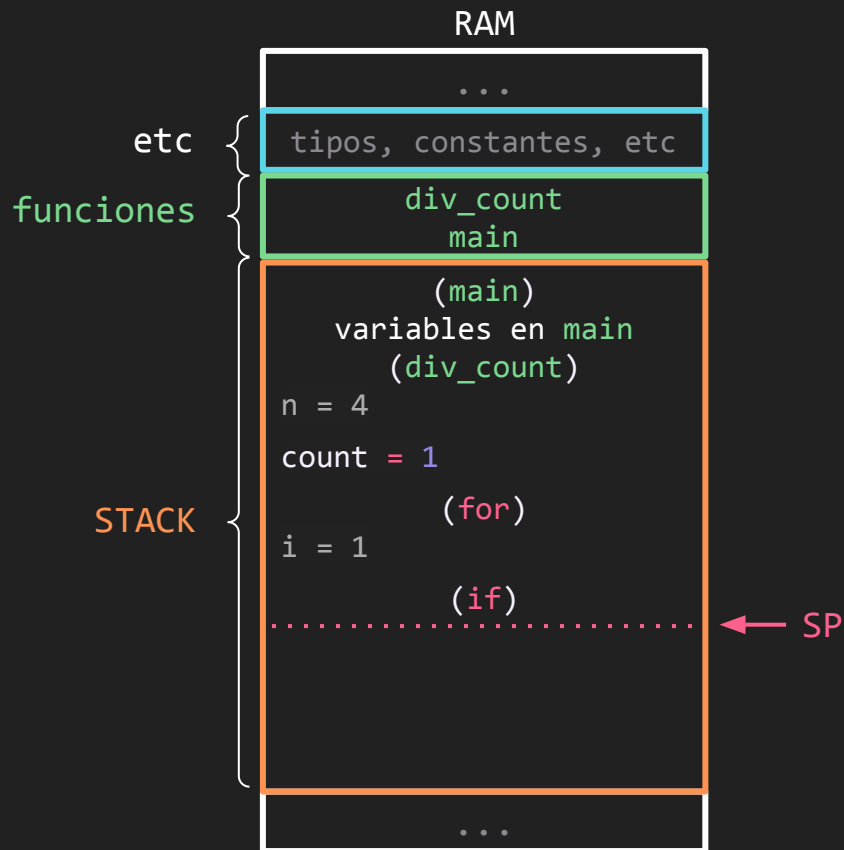
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



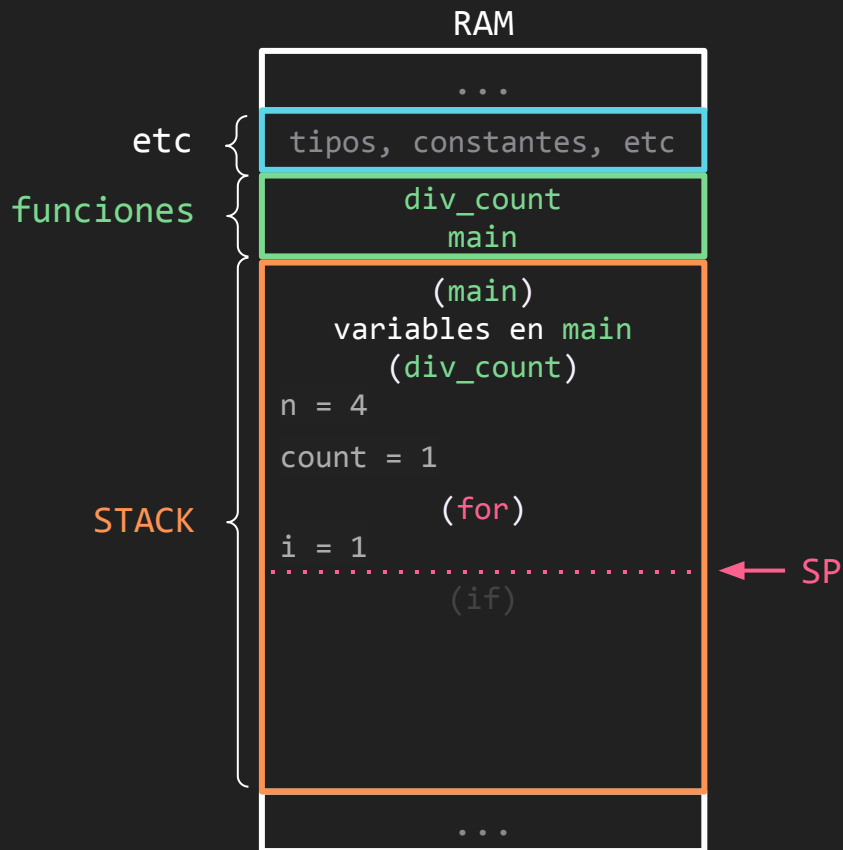
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



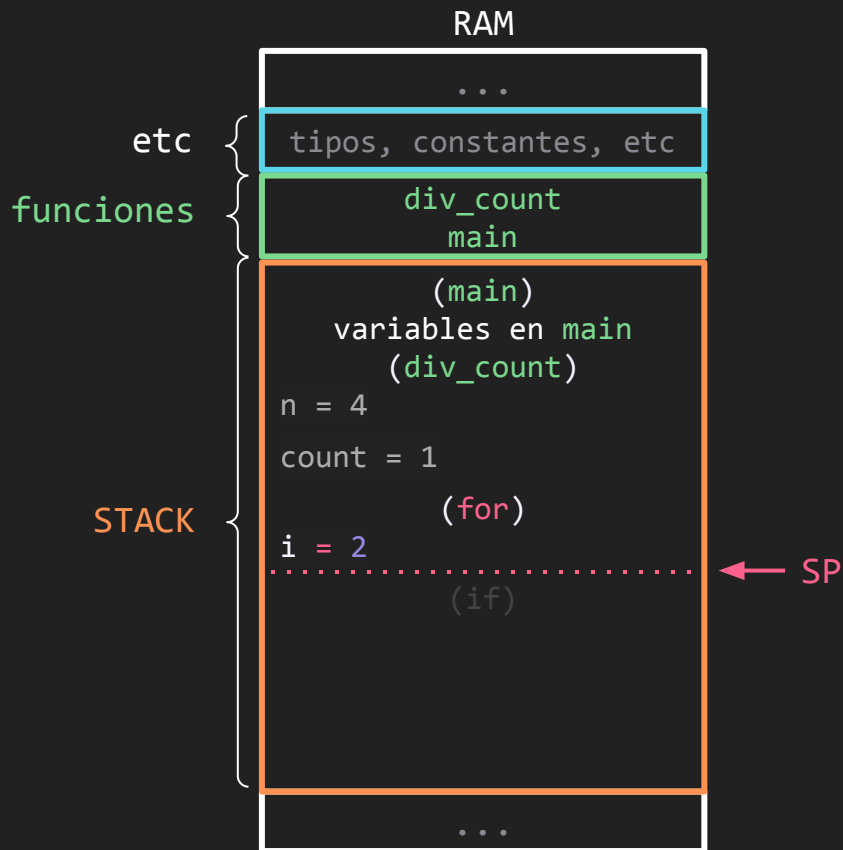
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



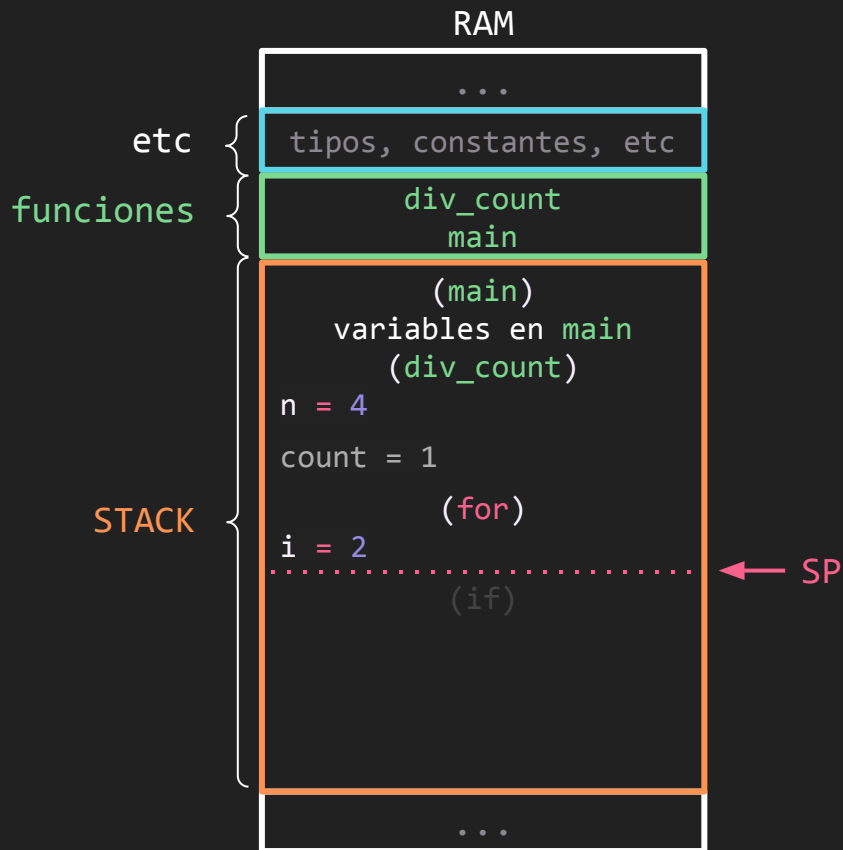
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



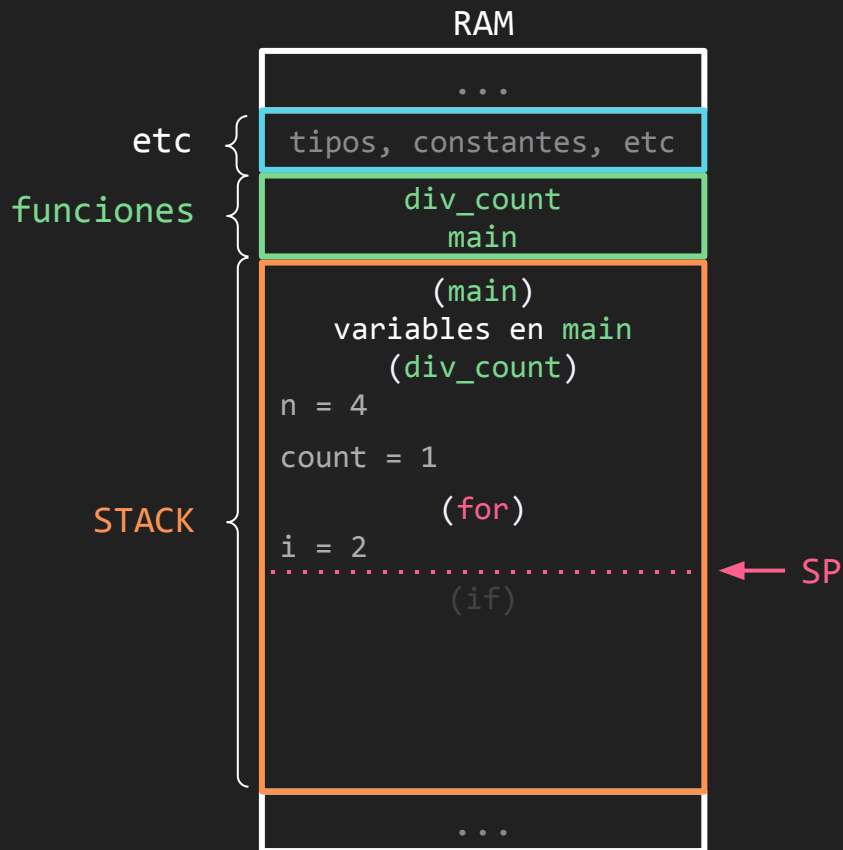
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



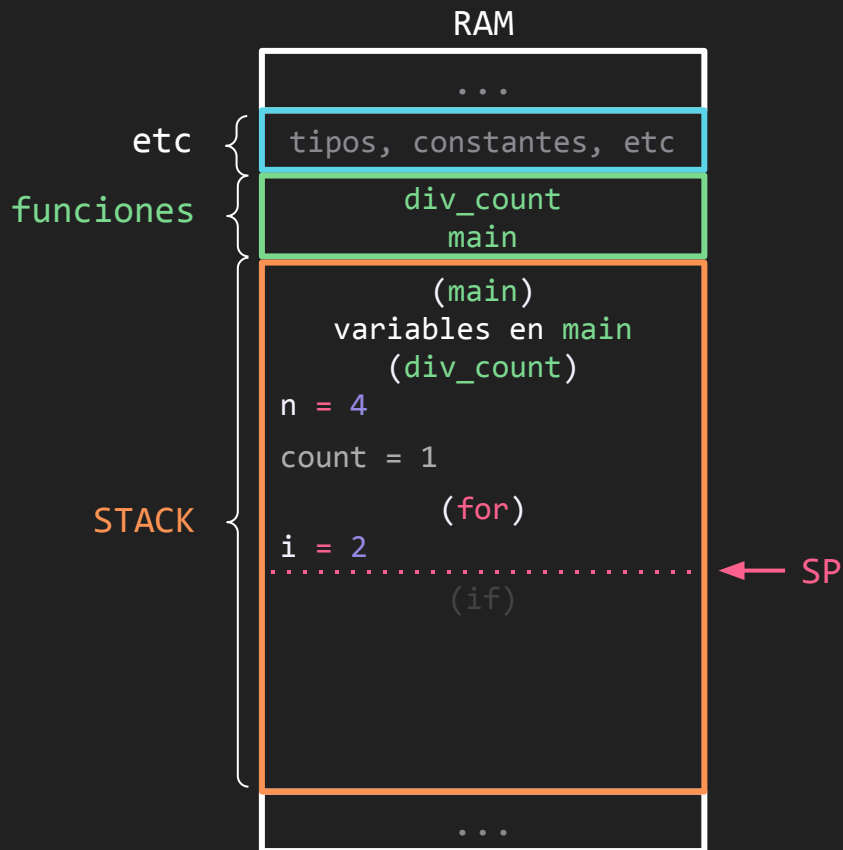
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



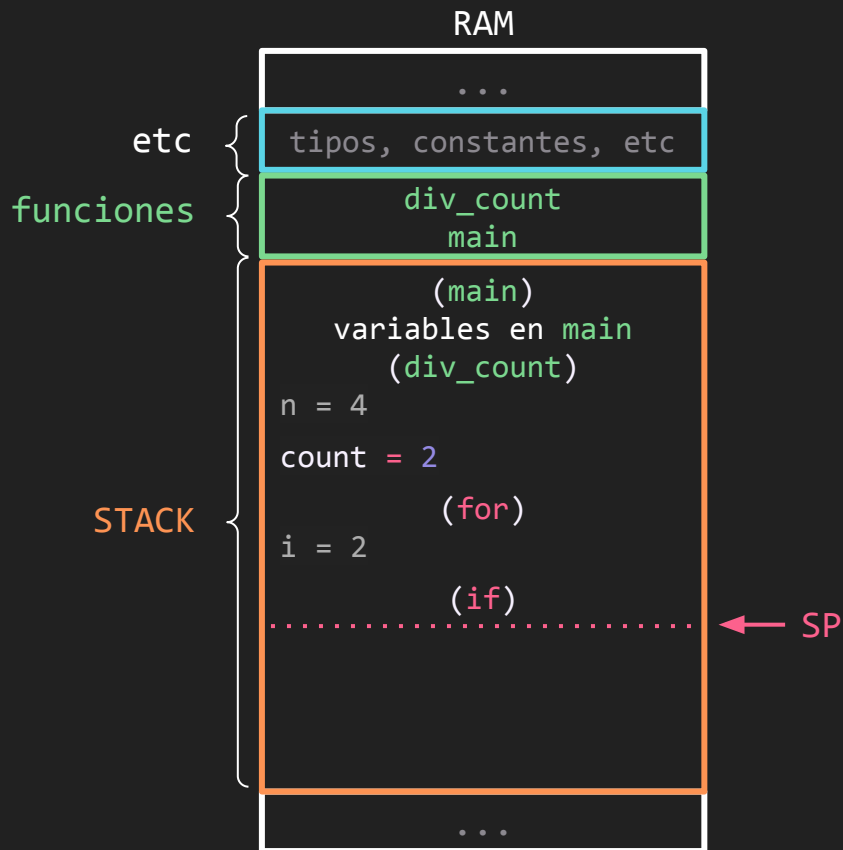
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



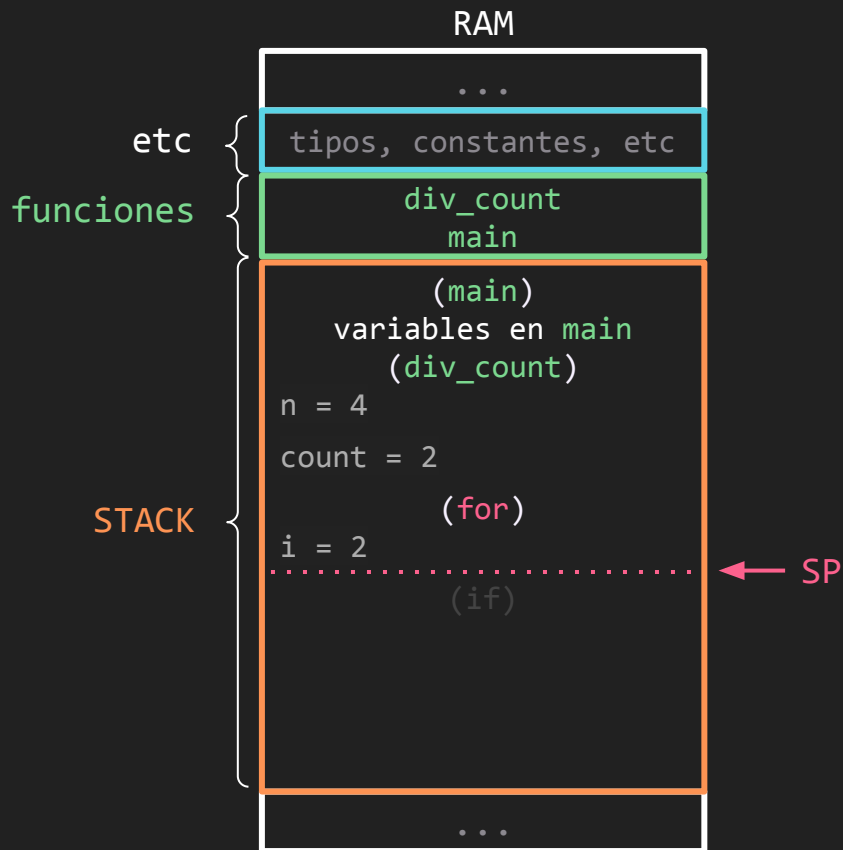
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



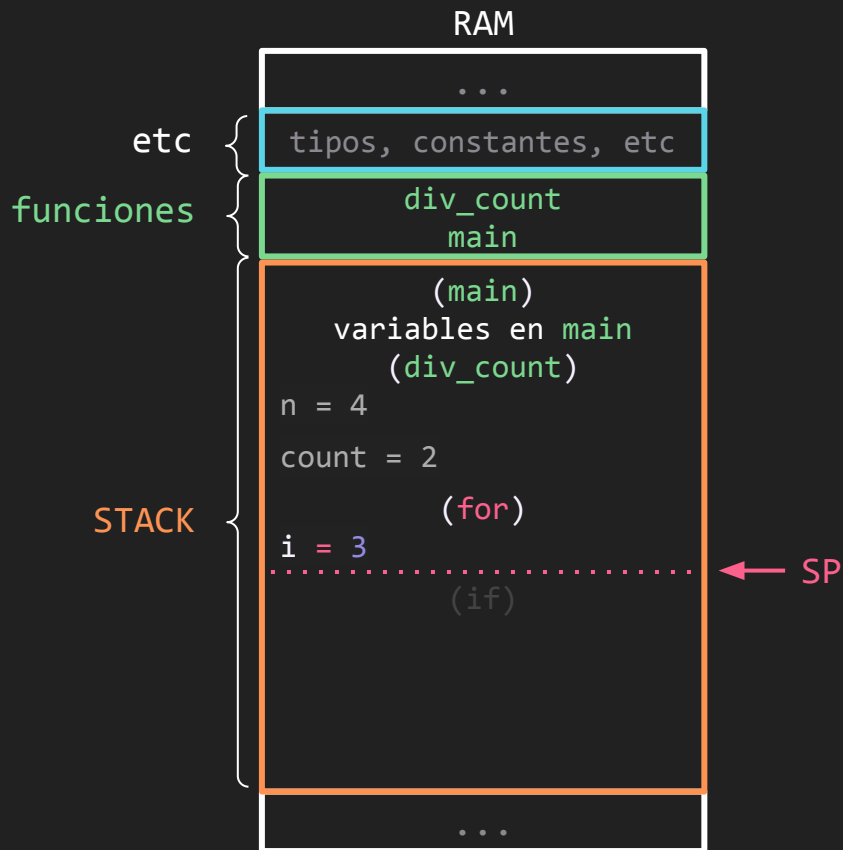
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



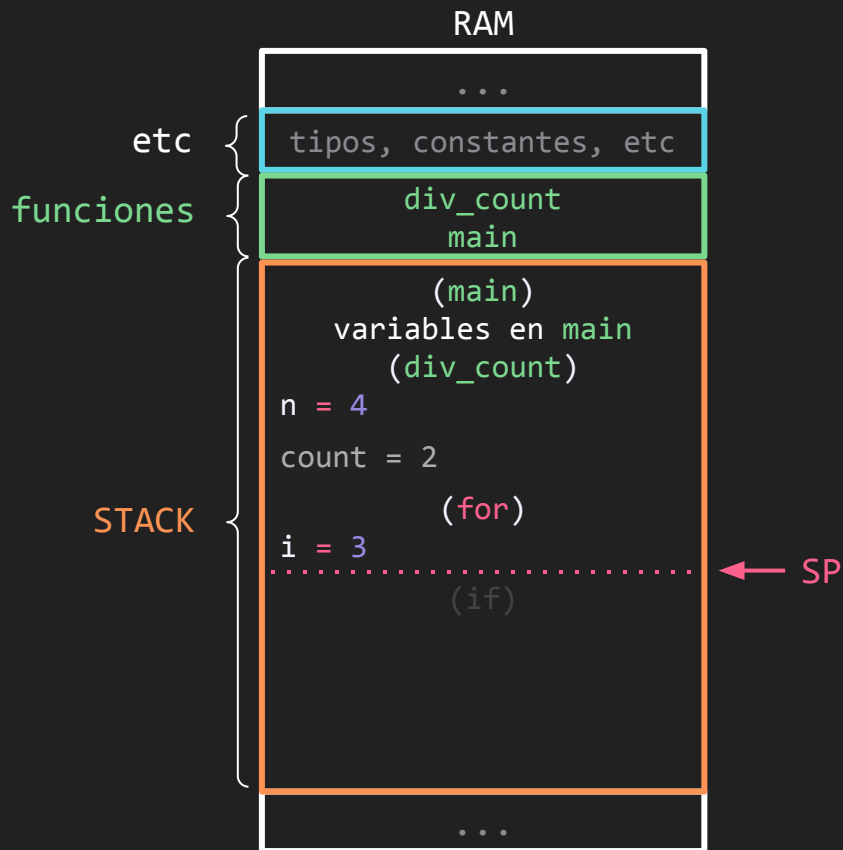
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



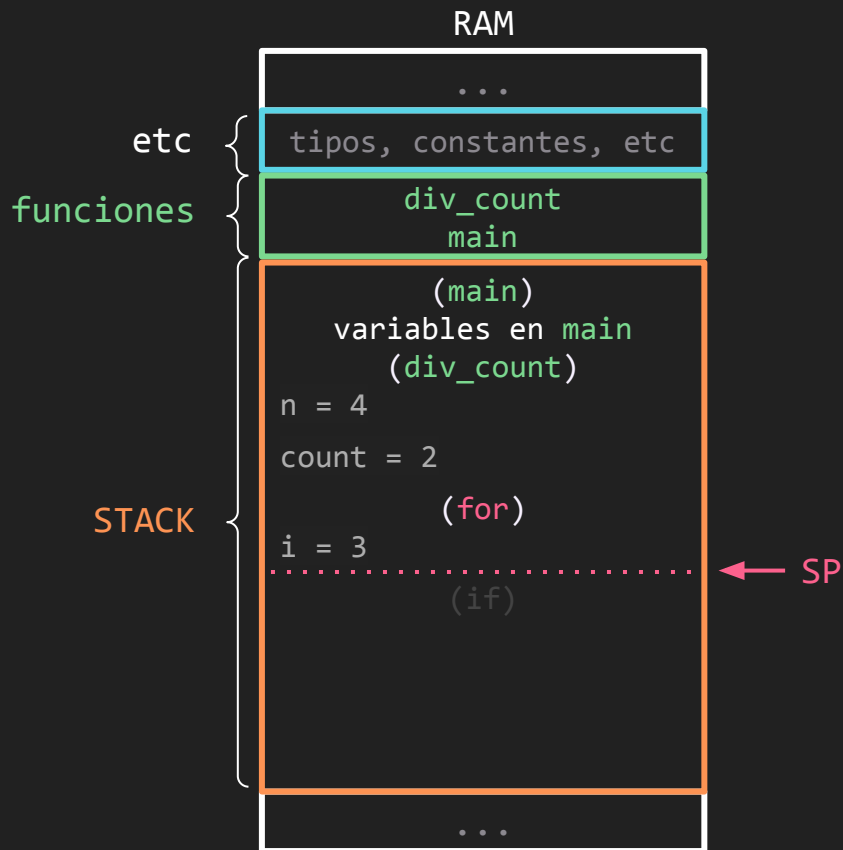
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



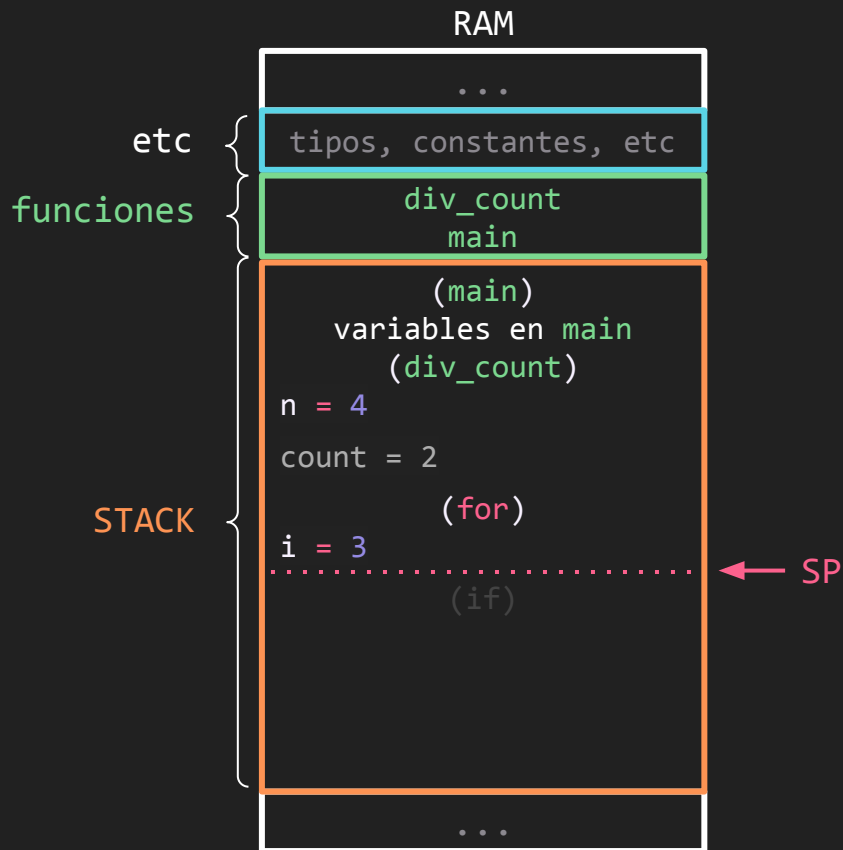
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



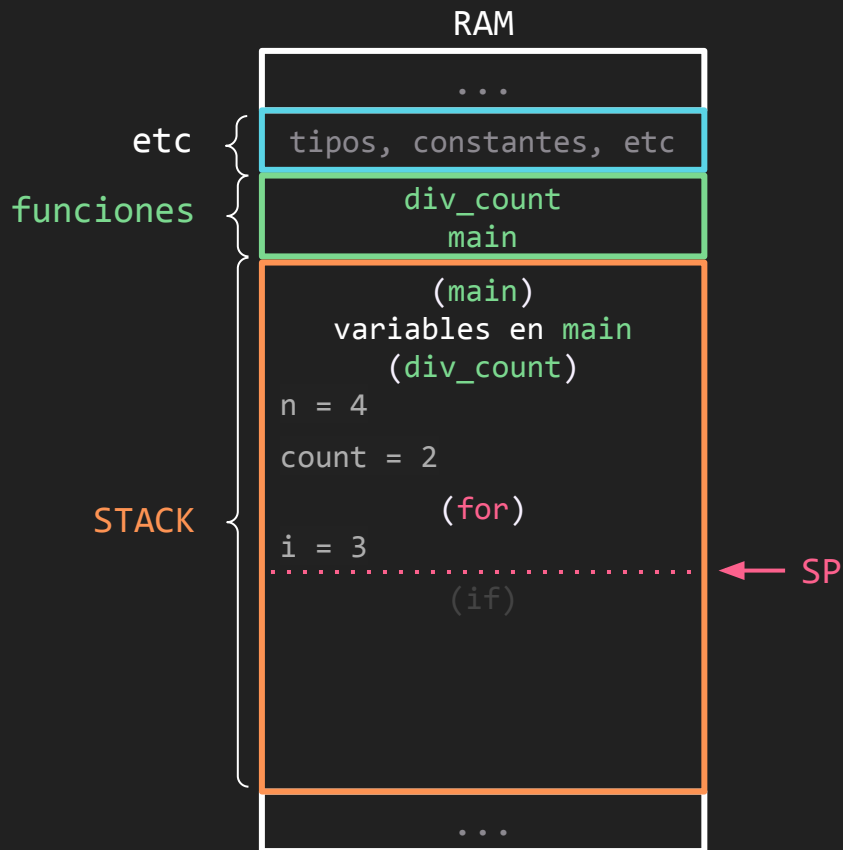
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



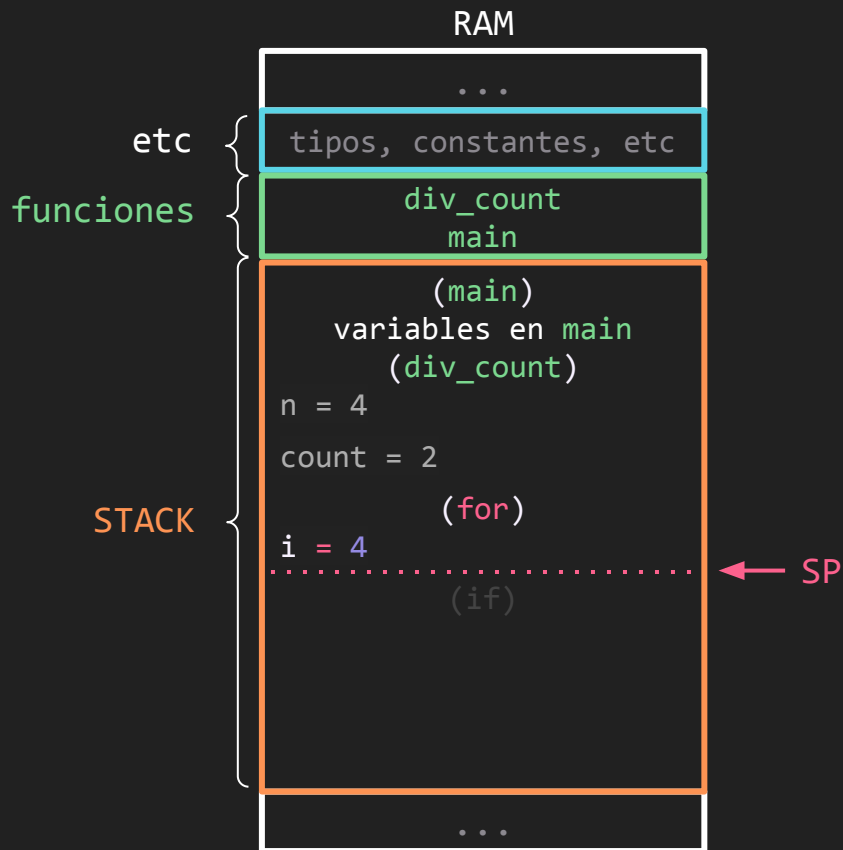
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



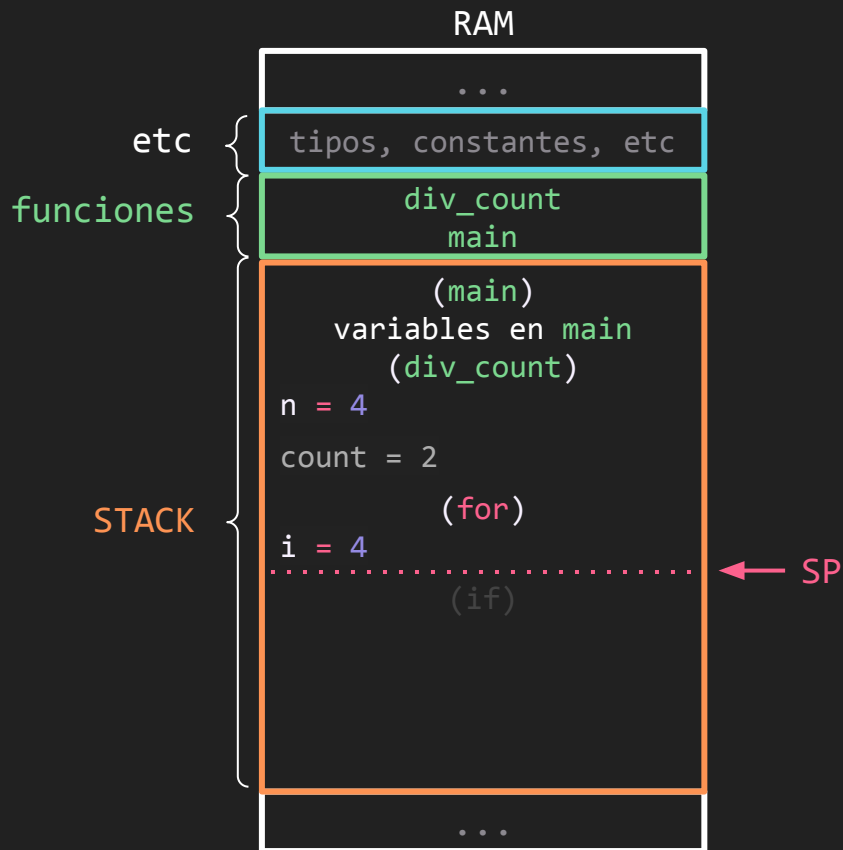
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



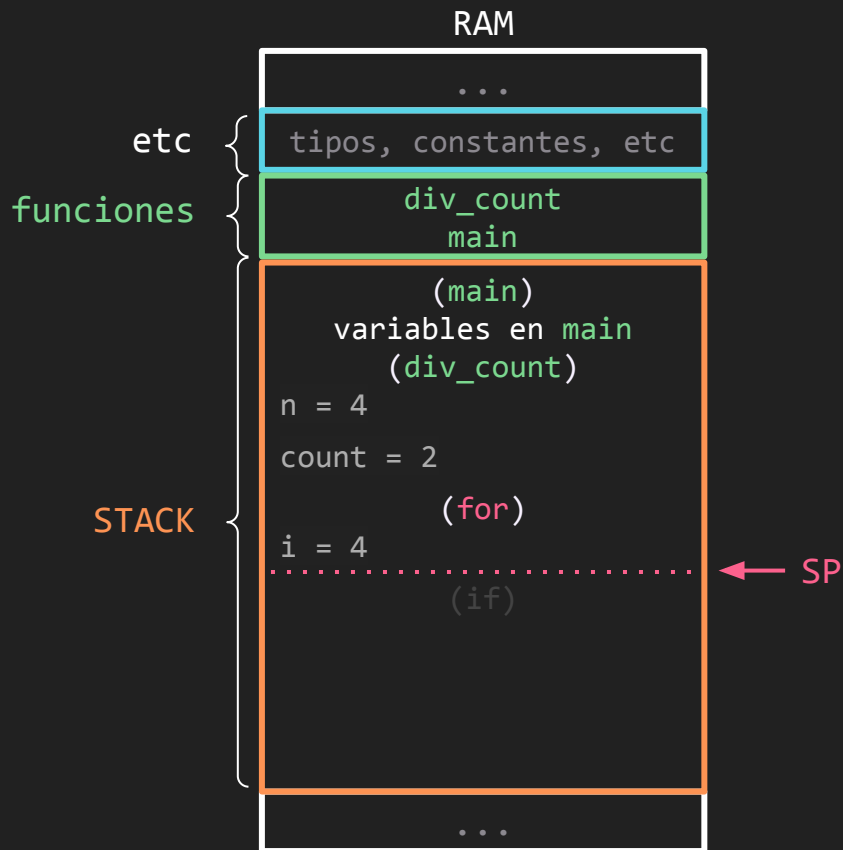
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



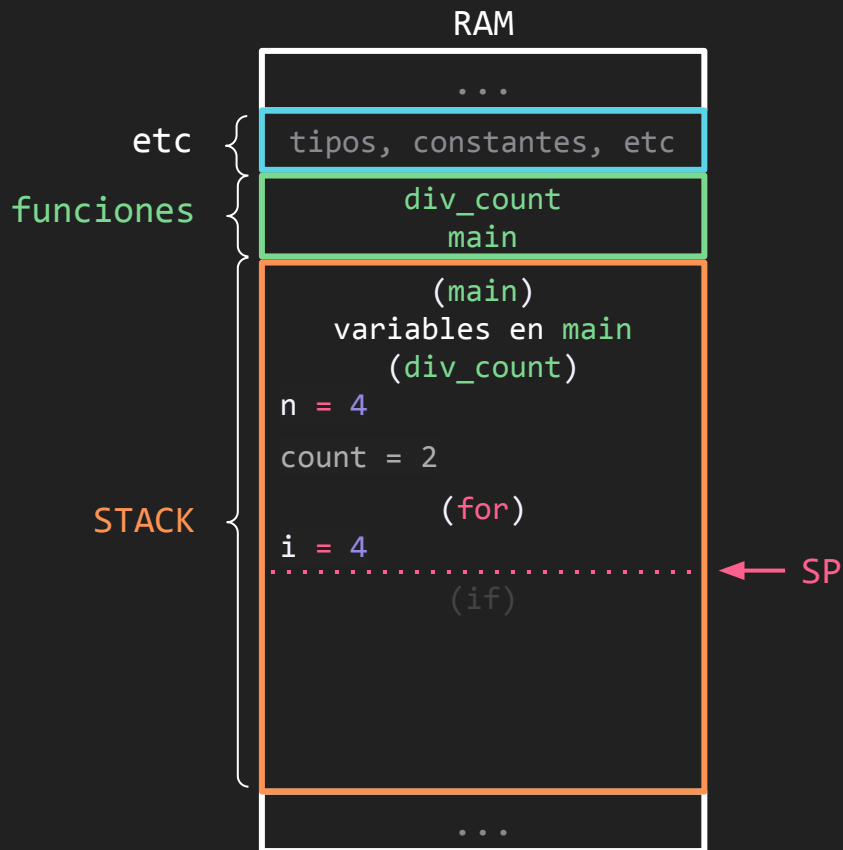
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



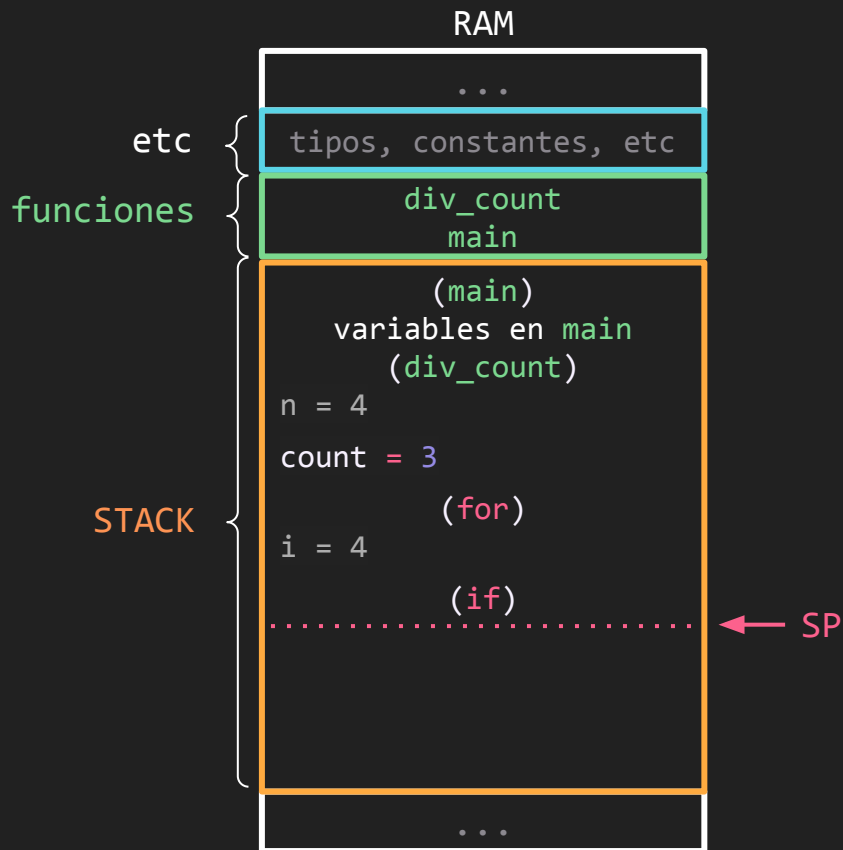
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



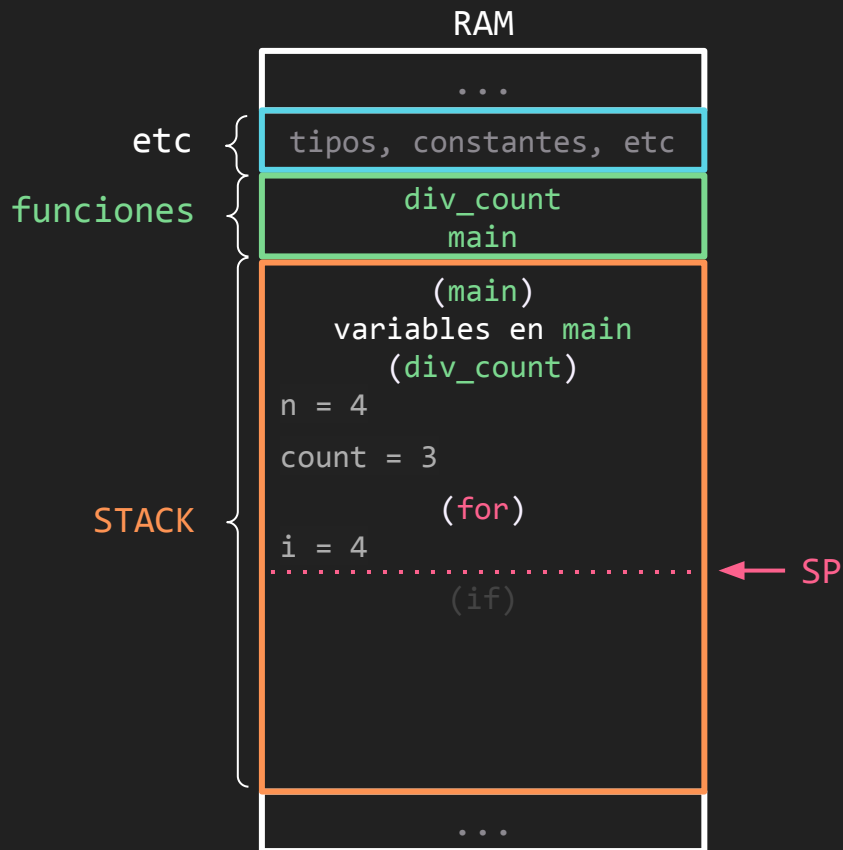
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



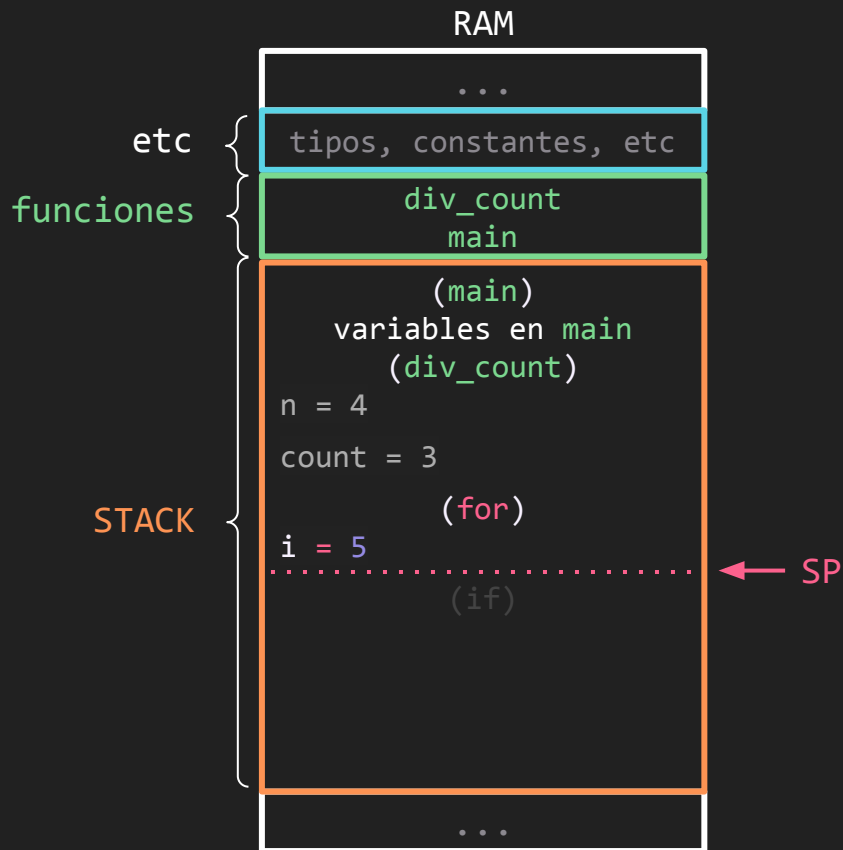
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



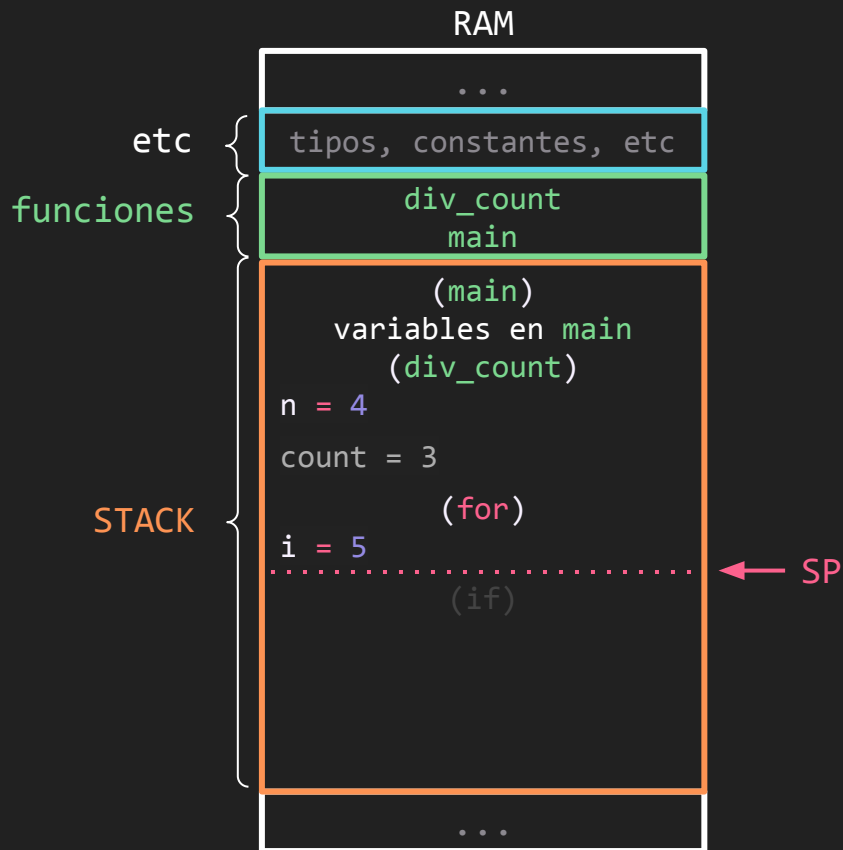
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



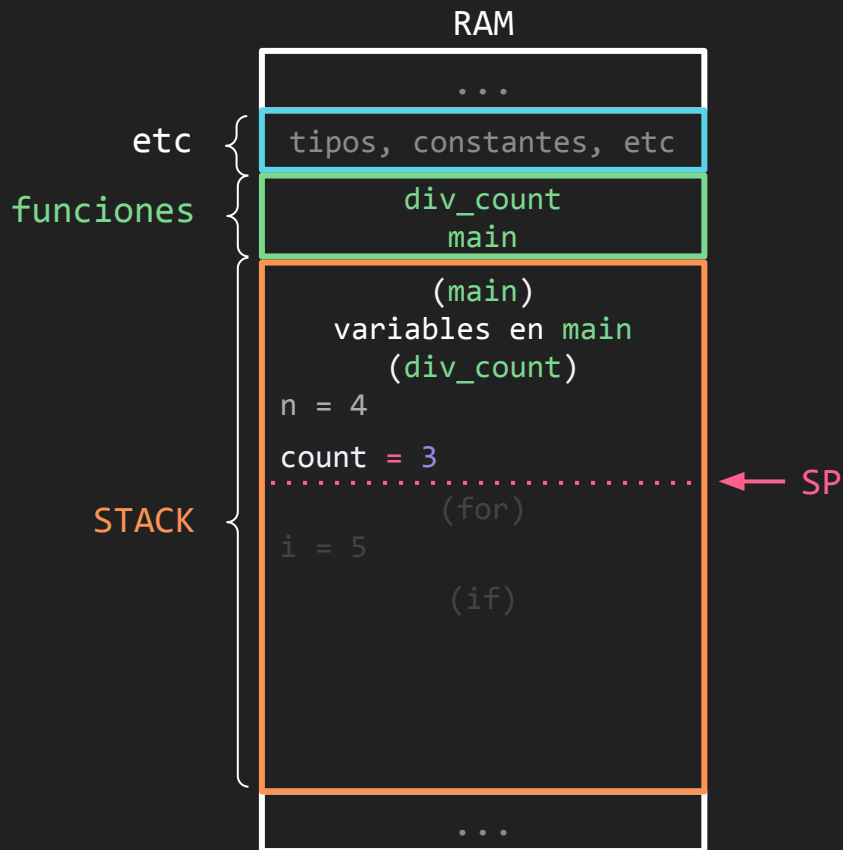
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



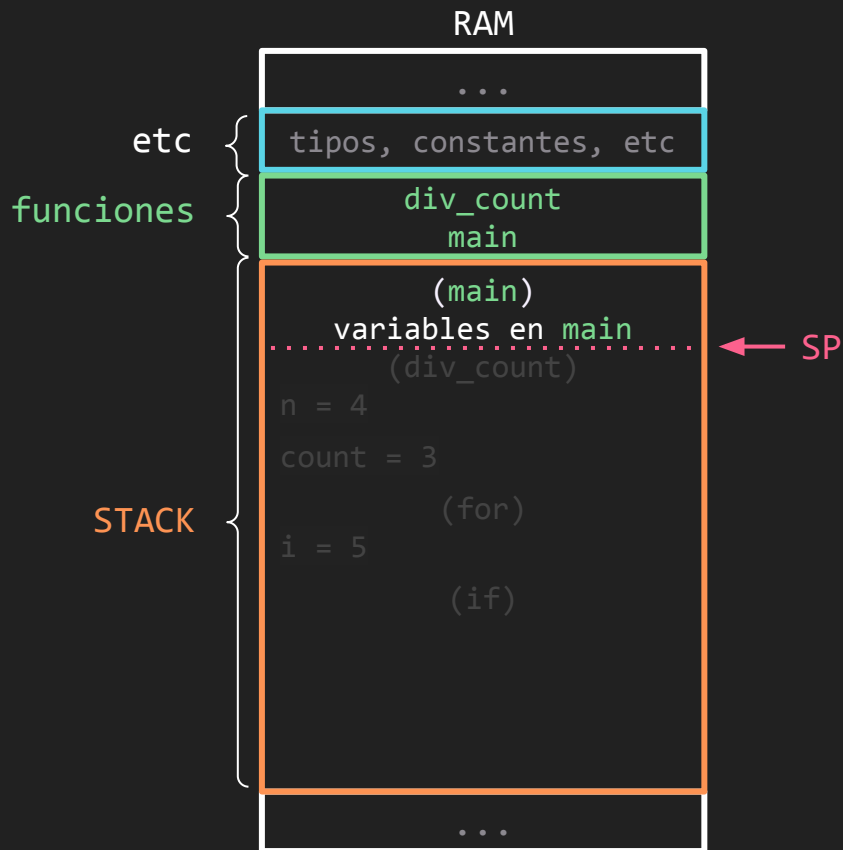
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



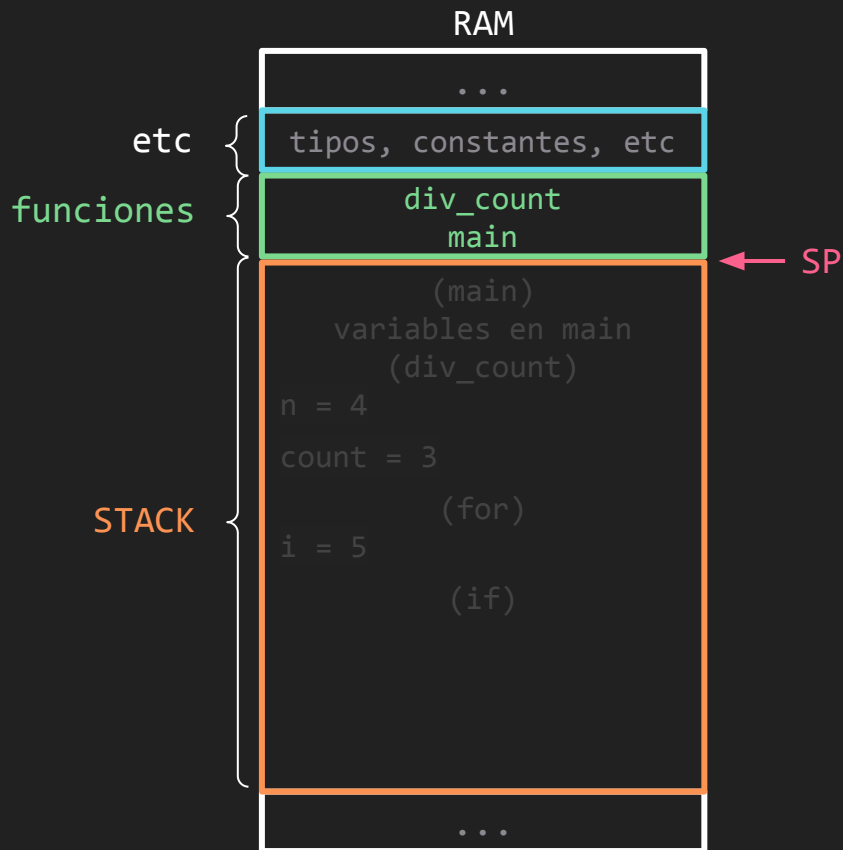
STACK de memoria



```
int div_count(int n)
{
    int count = 0;

    for(int i = 1; i <= n; i+=1)
    {
        if(n % i == 0) count += 1;
    }
    return count;
}

// Llama a div_count con n = 4
int main();
```



¡Muchas Gracias!



**"Why is
my function
not returning
anything?"**



**"Oh, I never
called it"**

With ♥ by @vichoeq & @KnowYourselves