```
int main()
{
	Node *linked_list;
	linked_list = 0;
	list_add_value(&linked_list, 1);
	list_add_value(&linked_list, 2);
	return 0;
}


Node* create_Node(int data)
{
	Node *new_node;
	new_node = (Node*)malloc(sizeof(Node));
	new_node->data = data;
	new_node->next = NULL;
	return new_node;
}


void list_add_value(Node **head, int value)
{
	Node **tmp, *new_node;
	new_node = create_Node(value);
	tmp = head;
	while ( (*tmp) )
	{
		tmp = &((*tmp)->next);
	}
	*tmp = new_node;
	(*tmp)->next = 0;
}
```

Para efectos gráficos, hice cuadritos que tienen arriba su dirección de memoria (si te interes la real, se puede calcular… te puedo enseñar si quieres, hoy no lo hice), debajo del cuadrito viene el nombre de la variable y dentro del cuadrito viene la información que contiene, igual en caso de apuntadores puse una flechita, lo que apunta la flechita y lo que viene dentro del cuadrito es equivalente, lo puse nomas por didáctica.

Del lado derecho te puse el cómo funciona realmente la memoria… las direcciones están MAL, pero lo demás si es así

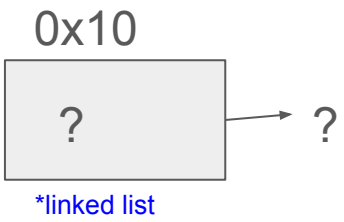
```
int main()
{
        Node *linked_list;
        linked_list = 0;
        list_add_value(&linked_list, 1);
        list_add_value(&linked_list, 2);
        return 0;
}
```

Memoria

Global 0x00-0x10

Main 0x10-0x20

Dirección	Dato
0x10	trash



DISCLAIMER:
Lo que está dentro de los cuadritos, es la información que contienen, las flechitas, son meramente cosas didácticas, pero en caso de apuntadores, lo que puse dentro del cuadrito y al lado de la flechita, es equivalente

```
int main()
{
        Node *linked_list;
        linked_list = 0;
        list_add_value(&linked_list, 1);
        list_add_value(&linked_list, 2);
        return 0;
}
```

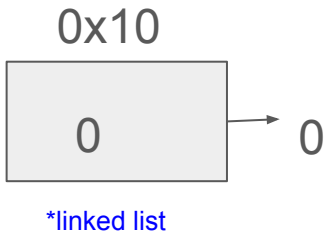
Algo que me dio duda aquí y podrías investigarlo, es qué pasa si el linked list no lo inicializas con 0, si dejas el linked_list así con las basura que traiga, qué pasa en el while?

Memoria

Global 0x00-0x10

Main 0x10-0x20

Dirección	Dato
0x10	0



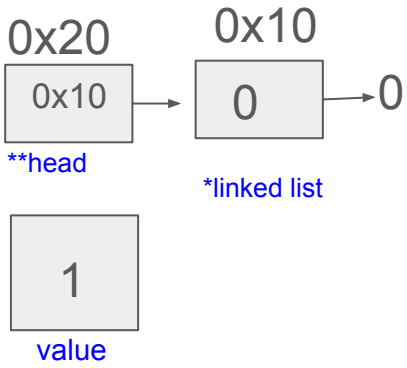
```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

Memoria

Global 0x00-0x10

Main 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

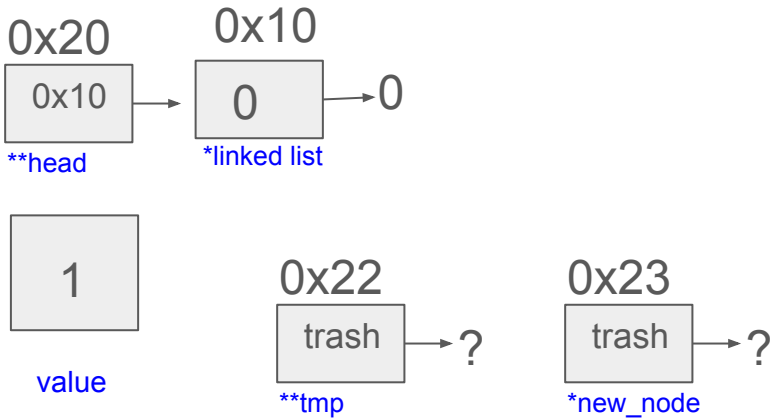
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	1
0x22	trash
0x23	trash



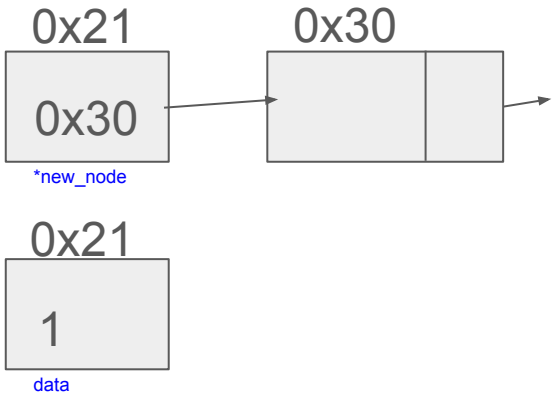
```
Node* create_Node(int data)
{
        Node *new_node;
        new_node = (Node*)malloc(sizeof(Node));
        new_node->data = data;
        new_node->next = NULL;
        return new_node;
}
```

Global 0x00-0x10

create_node 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	1
0x21	trash/trash
0x21	0x30



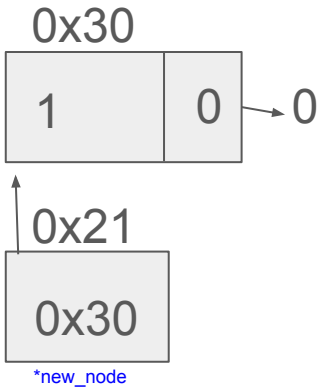
```
Node* create_Node(int data)
{
        Node *new_node;
        new_node = (Node*)malloc(sizeof(Node));
        new_node->data = data;
        new_node->next = NULL;
        return new_node;
}
```

Global 0x00-0x10

create_node0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	data
0x21	0x30



User	
0x30	1/0

RETURN -> 0x30

Regresa una dirección de memoria tipo Node… si me preguntas, te explico cómo sabe qué tipo de dirección de memoria es

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

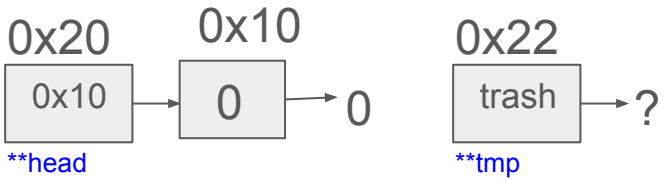
Memoria

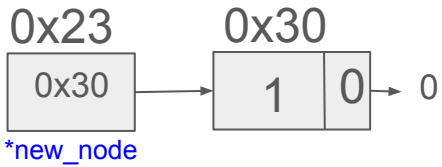
Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	1
0x22	trash
0x23	0x30





User	
0x30	1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

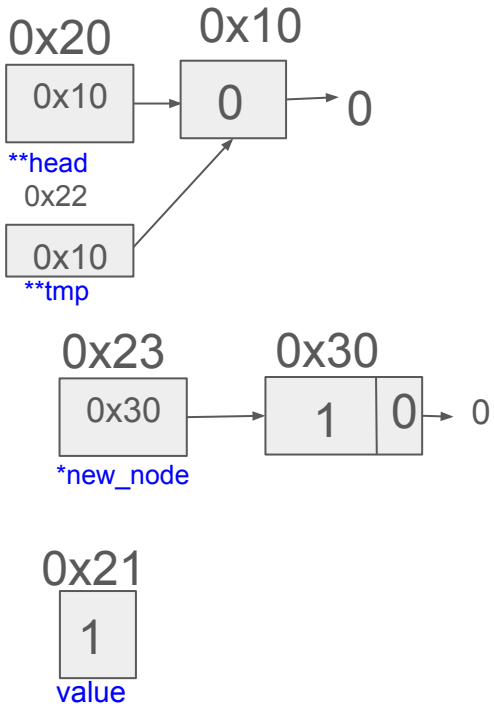
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x30



User	
0x30	1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

**tmp == 0x10
*tmp == 0

el while no se cumple

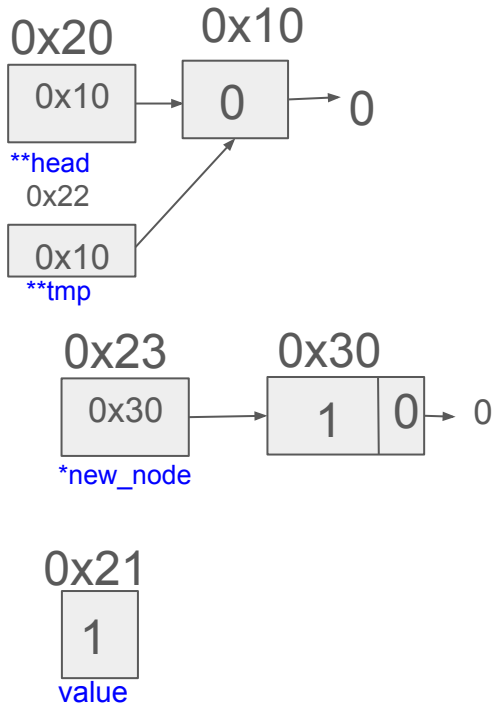
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x30



User
0x30 1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

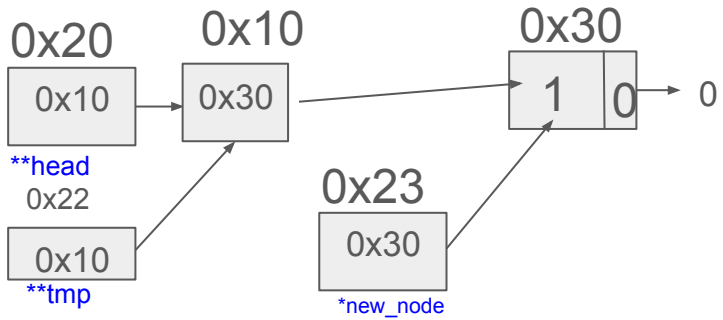
new_node = 0x30
recuerda que es un apuntador

Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x30

User

0x30	1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

*tmp = 0x10
(*tmp) = nodo en la dirección 0x30

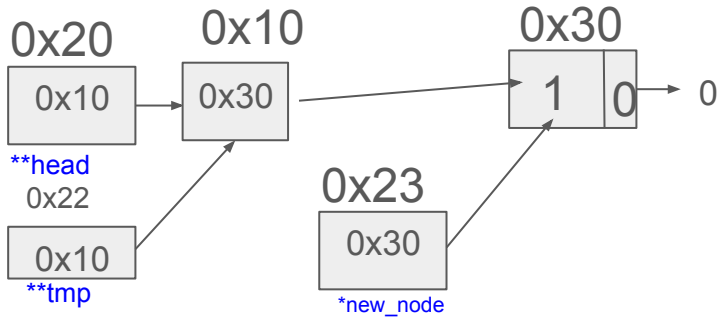
en este caso, esta instrucción no hizo nada, pero me gusta inicializar apuntadores en 0, es buena praxis

Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x30

User

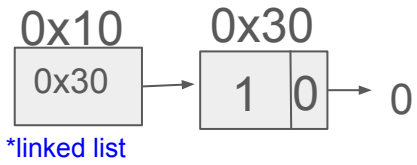
0x30	1/0

```
int main()
{
        Node *linked_list;
        linked_list = 0;
        list_add_value(&linked_list, 1);
        list_add_value(&linked_list, 2);
        return 0;
}
```

Memoria

Global 0x00-0x10

Main 0x10-0x20



Dirección	Dato
0x10	0x30

User	
0x30	1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

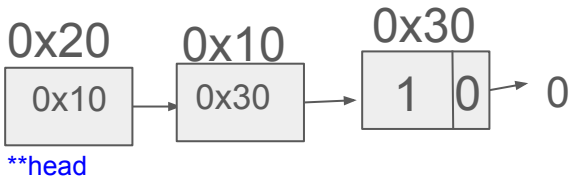
Memoria

Global 0x00-0x10

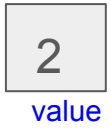
list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	2



User	
0x30	1/0



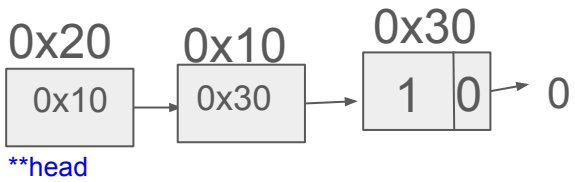
```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

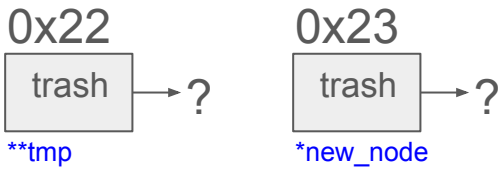
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40





Dirección	Dato
0x20	0x10
0x21	1
0x22	trash
0x23	trash

User	
0x30	1/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

Me salté el create_Node porque ya lo puse atras y es idéntico, solo obvio cambia la locación de memoria

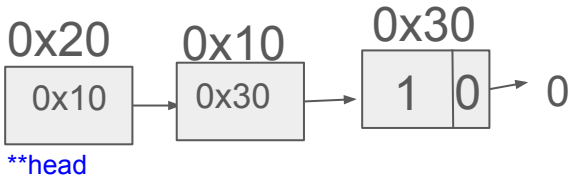
Memoria

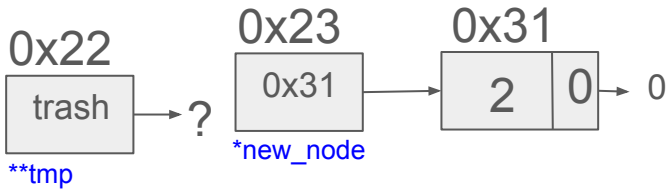
Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40

Dirección	Dato
0x20	0x10
0x21	1
0x22	trash
0x23	0x30





User	
0x30	1/0
0x31	2/0

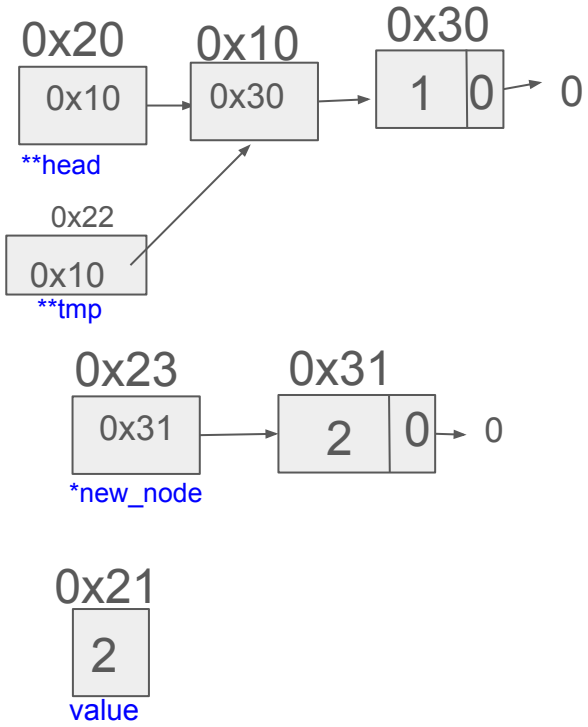
```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x30

User	
0x30	1/0
0x31	2/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

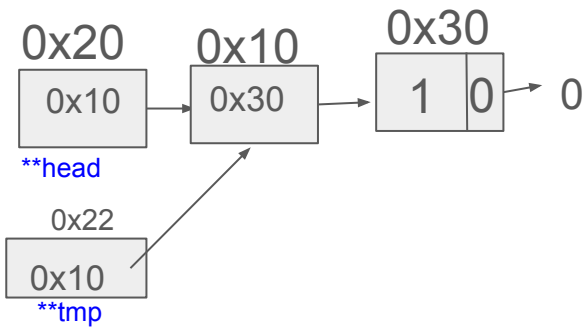
*tmp == 0x30
(*tmp)!=0, s.i entra

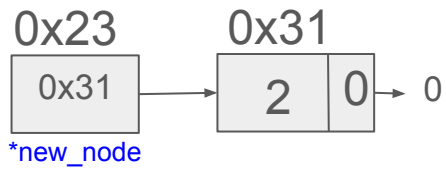
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40





Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x31

User	
0x30	1/0
0x31	2/0

```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

*tmp = 0x10
(*tmp) = se usa para poder acceder a sus parámetros
(*tmp)->next == 0
&(*tmp)->next == dirección del apuntador a nodo del nodo 0x30
tmp = dirección del next del 0x30

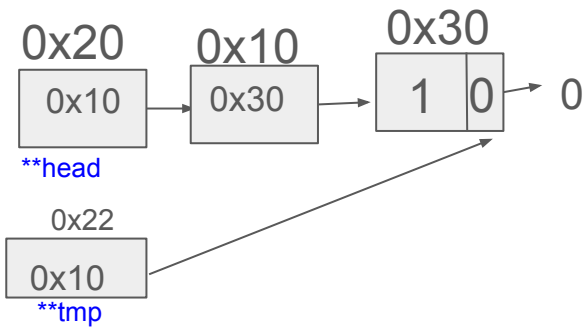
***Recuerda que el nodo está compuesto por
int data
nodo* ptr
Si hay dudas aquí, hablamos de cómo se conforma una struct… insisto, mis direcciones de memoria están ridículas, son didácticas

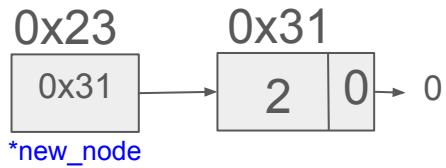
Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40





Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x31

User	
0x30	1/0
0x31	2/0

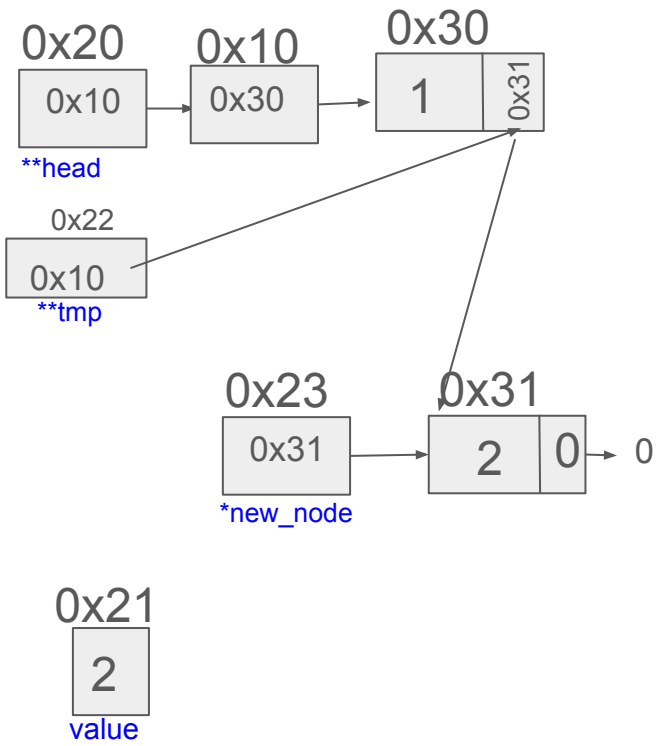
```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x31

User	
0x30	1/0
0x31	2/0

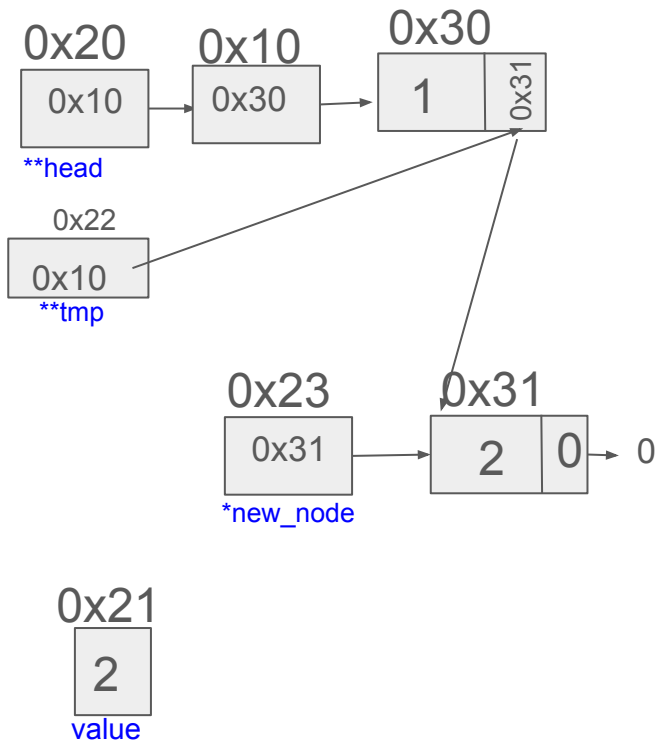
```
void list_add_value(Node **head, int value)
{
        Node **tmp, *new_node;
        new_node = create_Node(value);
        tmp = head;
        while ( (*tmp) )
        {
                tmp = &((*tmp)->next);
        }
        *tmp = new_node;
        (*tmp)->next = 0;
}
```

Memoria

Global 0x00-0x10

list_add_value 0x20-0x30

Usuario 0x30-0x40



Dirección	Dato
0x20	0x10
0x21	1
0x22	0x10
0x23	0x31

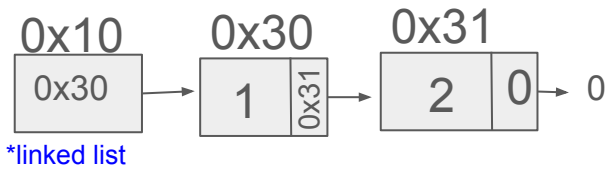
User	
0x30	1/0x31
0x31	2/0

```
int main()
{
        Node *linked_list;
        linked_list = 0;
        list_add_value(&linked_list, 1);
        list_add_value(&linked_list, 2);
        return 0;
}
```

Memoria

Global 0x00-0x10

Main 0x10-0x20



Dirección	Dato
0x10	0x30

User	
0x30	1/0x31
0x31	2/0