

```
1 // Implementa uma lista de números com um array de tamanho fixo
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7     // Lista de tamanho 3
8     int lista[3];
9
10    // Inicializa lista com números
11    lista[0] = 1;
12    lista[1] = 2;
13    lista[2] = 3;
14
15    // Imprime lista
16    for (int i = 0; i < 3; i++)
17    {
18        printf("%i\n", lista[i]);
19    }
20 }
```

```
1 // Implementa uma lista de números com um array de tamanho dinâmico
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main(void)
7 {
8     // Lista de tamanho 3
9     int *lista = malloc(3 * sizeof(int));
10    if (lista == NULL)
11    {
12        return 1;
13    }
14
15    // Inicializa lista de tamanho 3 com números
16    lista[0] = 1;
17    lista[1] = 2;
18    lista[2] = 3;
19
20    // Lista de tamanho 4
21    int *temporario = malloc(4 * sizeof(int));
22    if (temporario == NULL)
23    {
24        return 1;
25    }
26
27    // Copia lista de tamanho 3 para lista de tamanho 4
28    for (int i = 0; i < 3; i++)
29    {
30        temporario[i] = lista[i];
31    }
32
33    // Adiciona número à lista de tamanho 4
34    temporario[3] = 4;
35
36    // Libera lista de tamanho 3
37    free(lista);
38
39    // Grava lista de tamanho 4 no ponteiro *lista
40    lista = temporario;
41
42    // Imprime lista
43    for (int i = 0; i < 4; i++)
44    {
45        printf("%i\n", lista[i]);
```

```
46     }
47
48     // Libera lista
49     free(lista);
50 }
```

```
1 // Implementa uma lista de números com um array de tamanho dinâmico usando realloc
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main(void)
7 {
8     // Lista de tamanho 3
9     int *lista = malloc(3 * sizeof(int));
10    if (lista == NULL)
11    {
12        return 1;
13    }
14
15    // Inicializa lista de tamanho 3 com números
16    lista[0] = 1;
17    lista[1] = 2;
18    lista[2] = 3;
19
20    // Redimensiona lista para que tenha tamanho 4
21    int *temporario = realloc(lista, 4 * sizeof(int));
22    if (temporario == NULL)
23    {
24        return 1;
25    }
26    lista = temporario;
27
28    // Adiciona número à lista
29    lista[3] = 4;
30
31    // Imprime lista
32    for (int i = 0; i < 4; i++)
33    {
34        printf("%i\n", lista[i]);
35    }
36
37    // Libera lista
38    free(lista);
39 }
```

```
1 // Implementa uma lista de números usando uma lista encadeada
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 // Representa um nodo
7 typedef struct nodo
8 {
9     int numero;
10    struct nodo *proximo;
11 }
12 nodo;
13
14 int main(void)
15 {
16     // Lista de tamanho 0
17     nodo *lista = NULL;
18
19     // Adiciona número à lista
20     nodo *n = malloc(sizeof(nodo));
21     if (n == NULL)
22     {
23         return 1;
24     }
25     n->numero = 1;
26     n->proximo = NULL;
27     lista = n;
28
29     // Adiciona número à lista
30     n = malloc(sizeof(nodo));
31     if (n == NULL)
32     {
33         return 1;
34     }
35     n->numero = 2;
36     n->proximo = NULL;
37     lista->proximo = n;
38
39     // Adiciona número à lista
40     n = malloc(sizeof(nodo));
41     if (n == NULL)
42     {
43         return 1;
44     }
45     n->numero = 3;
```

```
46     n->proximo = NULL;
47     lista->proximo->proximo = n;
48
49     // Imprime lista
50     for (nodo *temporario = lista; temporario != NULL; temporario = temporario->proximo)
51     {
52         printf("%i\n", temporario->numero);
53     }
54
55     // Libera lista
56     while (lista != NULL)
57     {
58         nodo *temporario = lista->proximo;
59         free(lista);
60         lista = temporario;
61     }
62 }
```