

```
1 // Implements a list of numbers with an array of fixed size
2
3 #include <stdio.h>
4
5 int main(void)
6 {
7     // List of size 3
8     int list[3];
9
10    // Initialize list with numbers
11    list[0] = 1;
12    list[1] = 2;
13    list[2] = 3;
14
15    // Print list
16    for (int i = 0; i < 3; i++)
17    {
18        printf("%i\n", list[i]);
19    }
20 }
```

```
1 // Implements a list of numbers with an array of dynamic size
2 //
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main(void)
7 {
8     // List of size 3
9     int *list = malloc(3 * sizeof(int));
10    if (list == NULL)
11    {
12        return 1;
13    }
14
15    // Initialize list of size 3 with numbers
16    list[0] = 1;
17    list[1] = 2;
18    list[2] = 3;
19
20    // List of size 4
21    int *tmp = malloc(4 * sizeof(int));
22    if (tmp == NULL)
23    {
24        return 1;
25    }
26
27    // Copy list of size 3 into list of size 4
28    for (int i = 0; i < 3; i++)
29    {
30        tmp[i] = list[i];
31    }
32
33    // Add number to list of size 4
34    tmp[3] = 4;
35
36    // Free list of size 3
37    free(list);
38
39    // Remember list of size 4
40    list = tmp;
41
42    // Print list
43    for (int i = 0; i < 4; i++)
44    {
45        printf("%i\n", list[i]);
```

```
46     }
47
48     // Free list
49     free(list);
50 }
```

```
1 // Implements a list of numbers with an array of dynamic size using realloc
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 int main(void)
7 {
8     // List of size 3
9     int *list = malloc(3 * sizeof(int));
10    if (list == NULL)
11    {
12        return 1;
13    }
14
15    // Initialize list of size 3 with numbers
16    list[0] = 1;
17    list[1] = 2;
18    list[2] = 3;
19
20    // Resize list to be of size 4
21    int *tmp = realloc(list, 4 * sizeof(int));
22    if (tmp == NULL)
23    {
24        return 1;
25    }
26    list = tmp;
27
28    // Add number to list
29    list[3] = 4;
30
31    // Print list
32    for (int i = 0; i < 4; i++)
33    {
34        printf("%i\n", list[i]);
35    }
36
37    // Free list
38    free(list);
39 }
```

```
1 // Implements a list of numbers with linked list
2
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 // Represents a node
7 typedef struct node
8 {
9     int number;
10    struct node *next;
11 }
12 node;
13
14 int main(void)
15 {
16     // List of size 0
17     node *list = NULL;
18
19     // Add number to list
20     node *n = malloc(sizeof(node));
21     if (n == NULL)
22     {
23         return 1;
24     }
25     n->number = 1;
26     n->next = NULL;
27     list = n;
28
29     // Add number to list
30     n = malloc(sizeof(node));
31     if (n == NULL)
32     {
33         return 1;
34     }
35     n->number = 2;
36     n->next = NULL;
37     list->next = n;
38
39     // Add number to list
40     n = malloc(sizeof(node));
41     if (n == NULL)
42     {
43         return 1;
44     }
45     n->number = 3;
```

```
46     n->next = NULL;
47     list->next->next = n;
48
49     // Print list
50     for (node *tmp = list; tmp != NULL; tmp = tmp->next)
51     {
52         printf("%i\n", tmp->number);
53     }
54
55     // Free list
56     while (list != NULL)
57     {
58         node *tmp = list->next;
59         free(list);
60         list = tmp;
61     }
62 }
```