

```
33     for (int i = 0; i < 4; i++) {
34         printf("%d ", array2[i]);
35     }
36
37 }
38
39 printf("\n\n");
40
41 return 0;
42
43 }
```

Current numbers in array2 are: 3 1 1866658584 1

Current numbers in array3 are: 10 11 12 13

Please enter a number: 4

Current numbers in array2 are: 3 4 1866658584 1

Current numbers in array3 are: 10 11 12 13

Please enter a number: 5

Current numbers in array2 are: 3 4 5 1

Current numbers in array3 are: 10 11 12 13

Please enter a number: 6

Current numbers in array2 are: 3 4 5 6

Current numbers in array3 are: 10 11 12 13

Please enter a number:

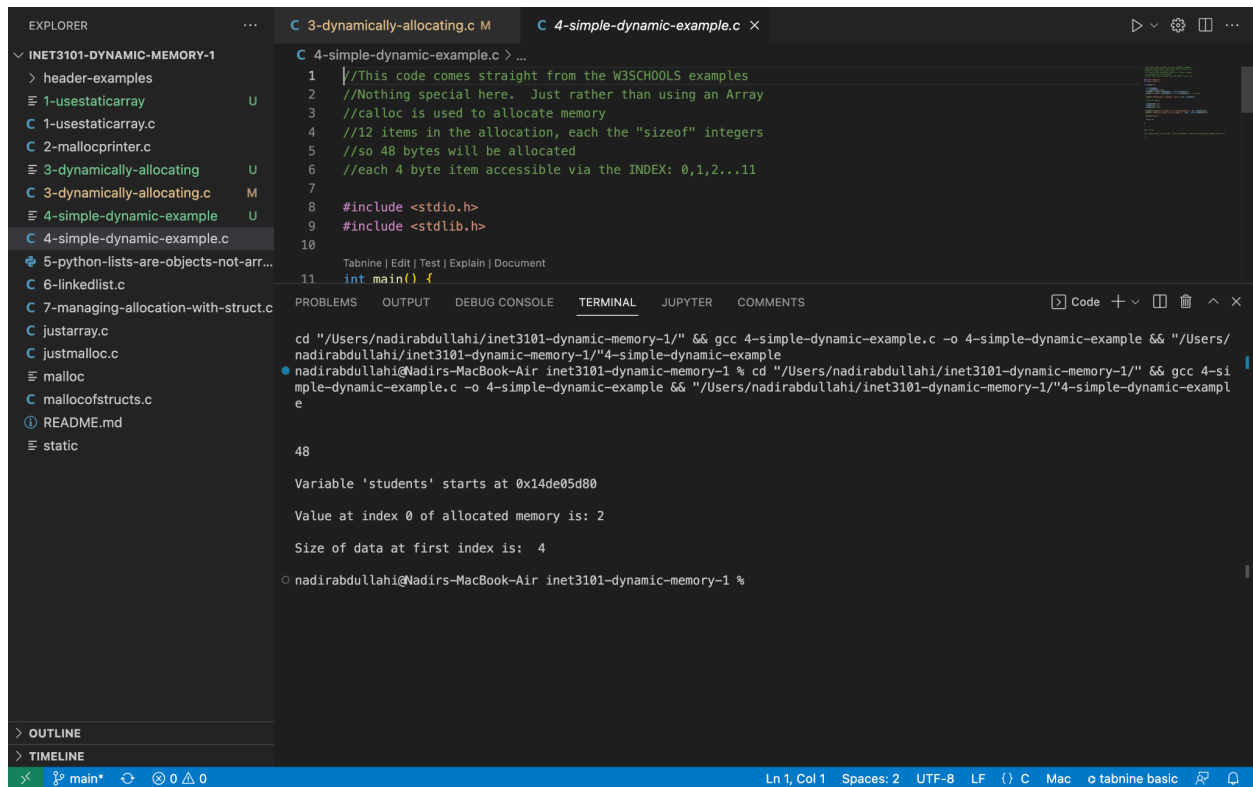
1.

3.

```
9
10 // "plumbing" is added to our main function so we can get a parameter passed from the command line
11 int main(int argc, char *argv[]) {
12     if (argc < 2) {
13         printf("Usage: %s <number_of_elements>\n", argv[0]);
14         return 1;
15     }
16
17     // based on how many elements the user wants, we set up the memory allocation
18     // even though this isn't a C array - it is being used like one, so we call it "array"
19     int num_elements = atoi(argv[1]);
20     int *array = (int *)malloc(num_elements * sizeof(int));
21
22     // now the user enters the data, type in a number, hit return/enter, enter another until done.
23     printf("Please enter %d numbers:\n", num_elements);
24
25     int i = 0;
26     while (i < num_elements) {
27         scanf("%d", &array[i]);
28         i++;
29     }
30
31     // then print out what was entered.
32     printf("Numbers in the array:\n");
33     for (int i = 0; i < num_elements; i++) {
34         printf("%d ", array[i]);
35     }
36     printf("\n");
37
38     // free up memory at the end. Allocated memory isn't automatically freed up by the OS like an C Array.
39     free(array);
40     return 0;
41 }
42
43 /*
44
```

In this case, it starts at the first index ($i = 0$) and keeps asking for numbers (`scanf("%d", &array[i])`), moving to the next position each time (`i++`).

4.



The screenshot shows a code editor with a file explorer on the left, a code editor in the center, and a terminal at the bottom. The file explorer shows a directory named 'INET3101-DYNAMIC-MEMORY-1' with several files, including '4-simple-dynamic-example.c'. The code editor shows the content of '4-simple-dynamic-example.c', which is a C program that dynamically allocates memory and prints the value at index 0. The terminal shows the command to compile and run the program, and the output, which is '48'.

```
1 //This code comes straight from the W3SCHOOLS examples
2 //Nothing special here. Just rather than using an Array
3 //calloc is used to allocate memory
4 //12 items in the allocation, each the "sizeof" integers
5 //so 48 bytes will be allocated
6 //each 4 byte item accessible via the INDEX: 0,1,2...11
7
8 #include <stdio.h>
9 #include <stdlib.h>
10
11 int main() {
12     int *array = (int *)calloc(12, sizeof(int));
13     printf("Enter 12 integers: ");
14     for (int i = 0; i < 12; i++) {
15         scanf("%d", &array[i]);
16     }
17     printf("Value at index 0 of allocated memory is: %d\n", array[0]);
18     return 0;
19 }
```

```
cd "/Users/nadirabdullahi/inet3101-dynamic-memory-1/" && gcc 4-simple-dynamic-example.c -o 4-simple-dynamic-example && "/Users/nadirabdullahi/inet3101-dynamic-memory-1/" 4-simple-dynamic-example
48
Variable 'students' starts at 0x14de05d80
Value at index 0 of allocated memory is: 2
Size of data at first index is: 4
nadirabdullahi@Nadirs-MacBook-Air inet3101-dynamic-memory-1 %
```

5. In Object-Oriented Programming (OOP), an object is a fundamental building block that represents a real-world entity or concept. It encapsulates data (attributes) and behaviors (methods) into a single unit, enabling modular and reusable code. Objects are instances of classes, which act as blueprints defining the structure and capabilities of the objects.

6. Linked lists helps us avoid using fixed sized arrays. With traditional arrays, you have to declare the size of it at the beginning, using a linked list you can use as you go. It also ensures you do not need to use memory reallocation because each node is stored separately and linked together.