Consider this double array:

Answers in next slide!

```
double example1[] = { 3.0, 4.5, 6.3, 9.0, 10.2 };
```

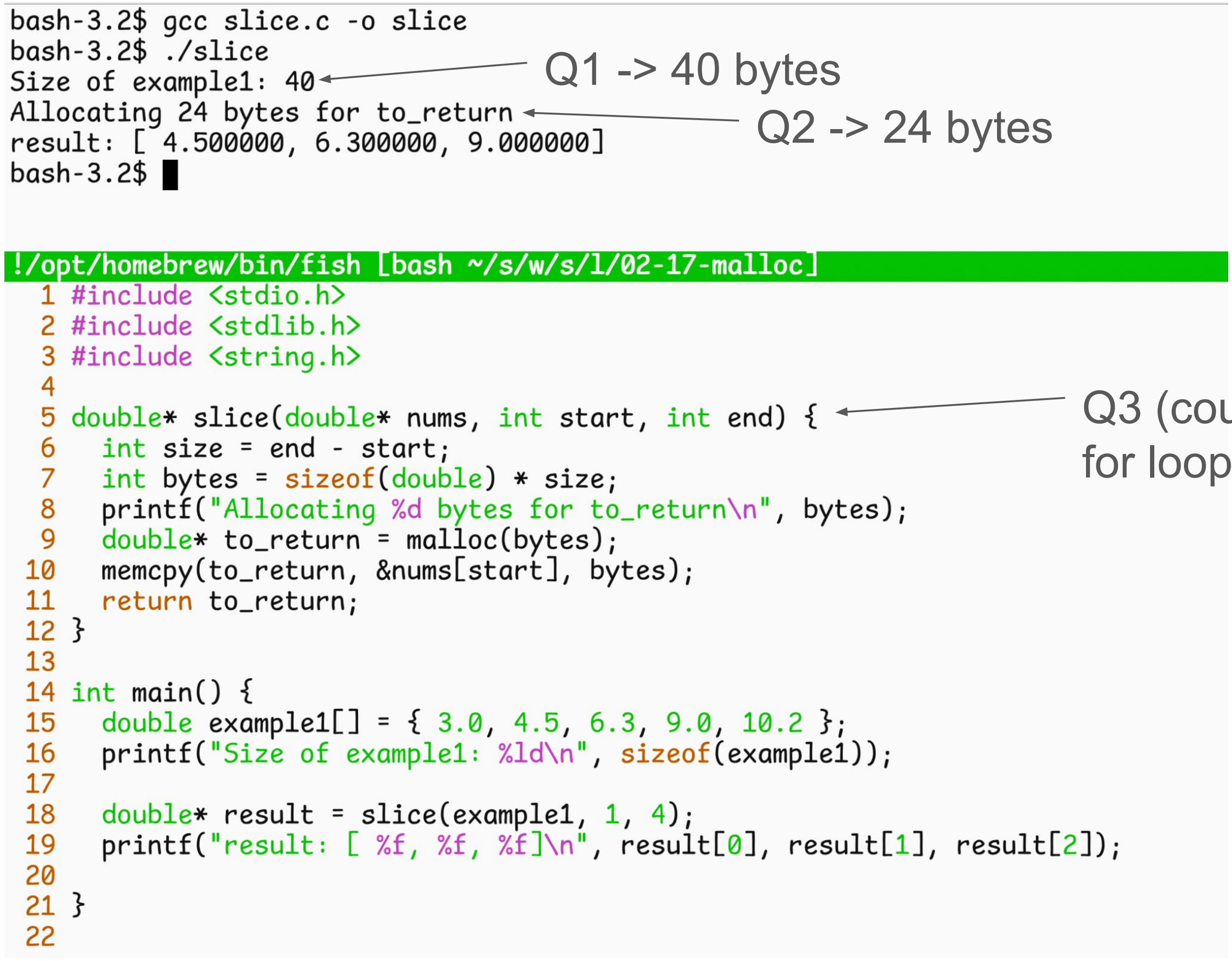
Q1: How many bytes long is it?
Q2: How many bytes are needed to store the elements at index 1, 2, and 3?

Write a function `slice` that takes an array of `double` called `nums`, a `start` index, and an `end` index, and returns a *new, heap-allocated array of `double`* that contains the elements from `nums` from indices `start` (inclusive) to `end` (exclusive)

Q3 (use the big box): Fill in this template to implement the function.

```
________ slice(___________, ____________, ________________) {

  double* to_return = malloc(___________________);






  return to_return;
}
```

Hey everyone! I'm home sick today, so the lecture format will be a little different – there will be a few videos and the course staff will run things!



```
bash-3.2$ gcc slice.c -o slice
bash-3.2$ ./slice
Size of example1: 40
Allocating 24 bytes for to_return
result: [ 4.500000, 6.300000, 9.000000]
bash-3.2$ █
```

Q1 -> 40 bytes

Q2 -> 24 bytes

```
!/opt/homebrew/bin/fish [bash ~/s/w/s/l/02-17-malloc]
```

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

double* slice(double* nums, int start, int end) {
  int size = end - start;
  int bytes = sizeof(double) * size;
  printf("Allocating %d bytes for to_return\n", bytes);
  double* to_return = malloc(bytes);
  memcpy(to_return, &nums[start], bytes);
  return to_return;
}

int main() {
  double example1[] = { 3.0, 4.5, 6.3, 9.0, 10.2 };
  printf("Size of example1: %ld\n", sizeof(example1));

  double* result = slice(example1, 1, 4);
  printf("result: [ %f, %f, %f]\n", result[0], result[1], result[2]);

}

```

Q3 (could also use a for loop)