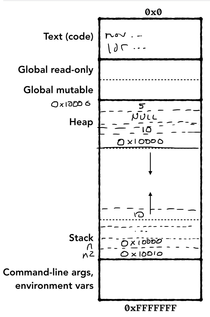
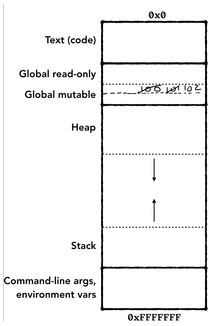



```
struct Node {
  int value;
  struct Node* next;
};

typedef struct Node Node;

Node* make_node(int v, Node* n) {
  Node* new_node = malloc(sizeof(Node));
  new_node->value = v;
  new_node->next = n;
}

int main() {
  Node* n = make_node(5, NULL);
  Node* n2 = make_node(10, n);
}
```



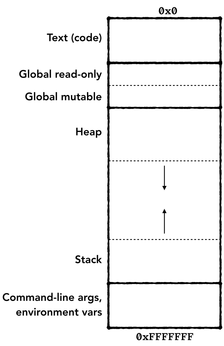

```
int global_var = 100;

void f() {
  global_var += 1;
}

int main() {
  f();
  f();
  printf("%d\n", global_var);
}
```

What does it print?

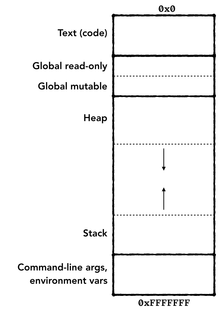
A. 100
B. 101
C. 102
D. Error



A tricky line of code:

```
int main() {
  char* my_str = "asdf";
  my_str[0] = 'z';
}
```

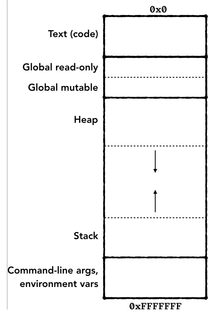
Where does the "asdf" get stored?



A tricky line of code:

```
int main() {
  char* my_str = "asdf";
  my_str[0] = 'z';
}
```

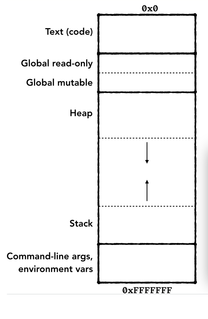
Where does the "asdf" get stored?



A useful trick:

```
// strdup makes a copy of a string
// by using malloc()
int main() {
  char* my_str = strdup("asdf");
  my_str[0] = 'z';
}
```

Where does the "asdf" get stored?



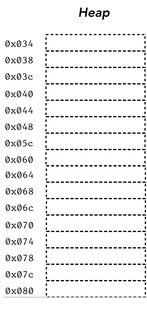
Seeing command-line arguments

```
#include <stdio.h>

int main(int argc, char** args) {
  int i;
  for(i = 0; i < argc; i += 1) {
    printf("%s\n", args[i]);
  }
  return 0;
}
```

```
$ gcc -o args args.c
$ ./args one two three
./args
one
two
three
$
```

Heap




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

int main(int argc, char** args) {
  int size = 10;
  int* some_nums = malloc(sizeof(int) * size);
  int* more_nums = malloc(sizeof(int) * 3);
  more_nums[0] = 0;
  some_nums[12] = 1;
  printf("%d\n", some_nums[12]);
  if(more_nums[0] > 0) {
    spend_money();
  }
}
```