Today's plan

- Class interactions
  - Q & A from last time
  - Group discussion on Ex 4-1
  - Recap the concepts
  - Quiz
- Class exercises
  - Ex 4-2

Q & A

- I've seen `**ptr` before. What does this mean?

Answer: It's a pointer-to-pointer. If you want to change the value of a pointer in a function, you will need to pass this pointer by another pointer. In this case, you will use a pointer-to-pointer. e.g.

```
1  void initArray(int **a, int asize) {
2          if (*a) free(*a);
3          *a = malloc(sizeof(int) * asize);
4  }
5  int main() {
6          int *array = NULL;
7          printf("Enter the array size you need:");
8          int arraysize = 0;
9          scanf("%d", &arraysize);
10         initArray(&array, arraysize);
11         ...
12 }
```

We need to pass the address of `array` to `initArray`, so that we can change its value inside the function. We use a pointer-to-pointer for that.

Q & A

- Is there a way to pull from the public repo without typing in our password too! The steps for ssh seem to only for the private one.

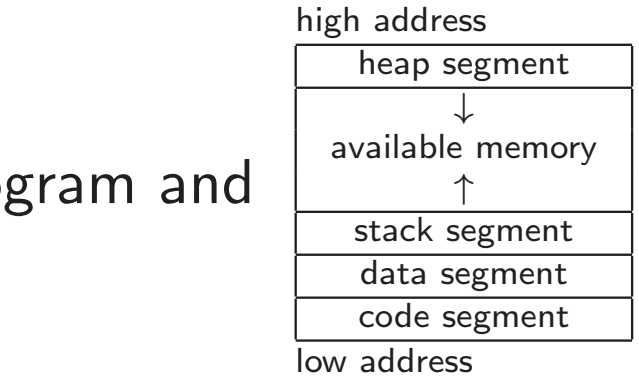
Answer: It should work for the public repo as well. Check `.git/config` to see if you have changed it to use git:// instead of https://.

Ex 4-1

Group discussion

- Share your implementation in your group and discuss these questions:
- How do you `scanf` the input with formats MM/DD/YYYY?
- How do you declare and initialize an array of strings for the names of the months?
- Will it work if pass the parameters by value into `getdate`, instead of by pointers?

Recap - dynamic memory allocation



- A little bit about memory of a running program and segmentation fault:
  - Code segment: the machine instructions
  - Data segment: static and global variables
  - Stack segment: parameters, non-static variables (known at compile time, statically allocated)
  - Heap segment: dynamic variables (known at run-time, dynamically allocated)
  - Between stack and heap are available memory.
- Invalid read/write, you are de-referencing a pointer that is pointing to unallocated memory.
- De-reference a pointer that is pointing to a "wrong" segment gives you the segmentation fault error!

Recap - dynamic memory allocation

- `malloc`: allocate dynamic memory.
- `calloc`: same as `malloc` and initialize values to zeros.
- `realloc`: reallocate memory (i.e. resize). Note, resulting pointer (memory start address) may change.
- `free`: deallocate the memory.
- We must deallocate what we allocated; otherwise, memory leak!
- Minimal example:

```c
#include <stdlib.h>

int main() {
    // allocate dynamic memory
    float *ptr = malloc(sizeof(float) * 100);
    for (int i = 0; i < 100; ++i) ptr[i] = i;

    // deallocate the memory
    free(ptr);

    return 0;
}
```

Recap - Valgrind

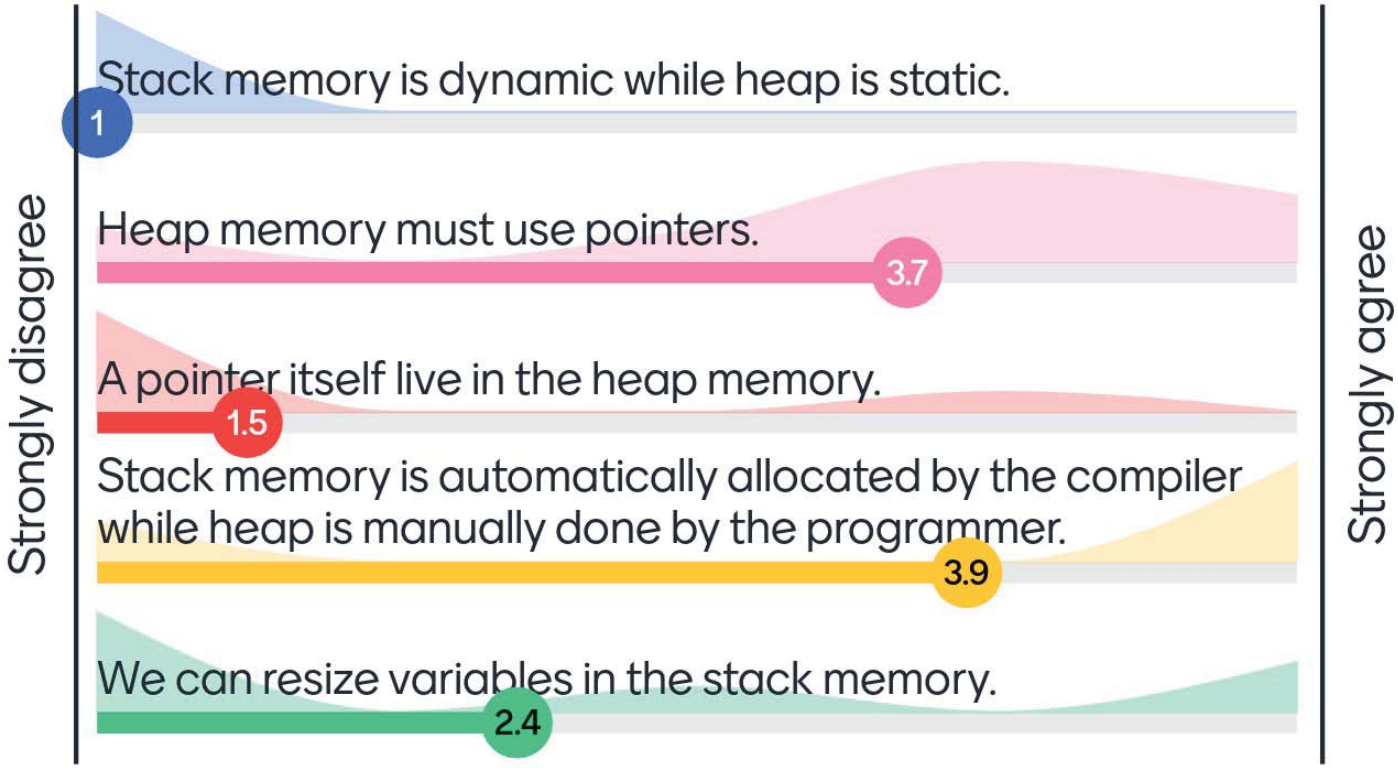
- Memory problems:
  - Invalid read/write: de-referencing a pointer that points to unallocated memory
  - Segmentation fault: de-referencing a pointer that points to a wrong "segment"
  - Memory leak: allocated memory but not freed
- Segmentation fault is easy to observe as it crashes your program.
- For invalid read/write and memory leak, we use `valgrind`
  - require debug symbols, i.e. compile program with '-g' flag
  - `valgrind --leak-check=full <program> <args ...>`

Quiz

Quiz!

What are true about stack and heap memory?

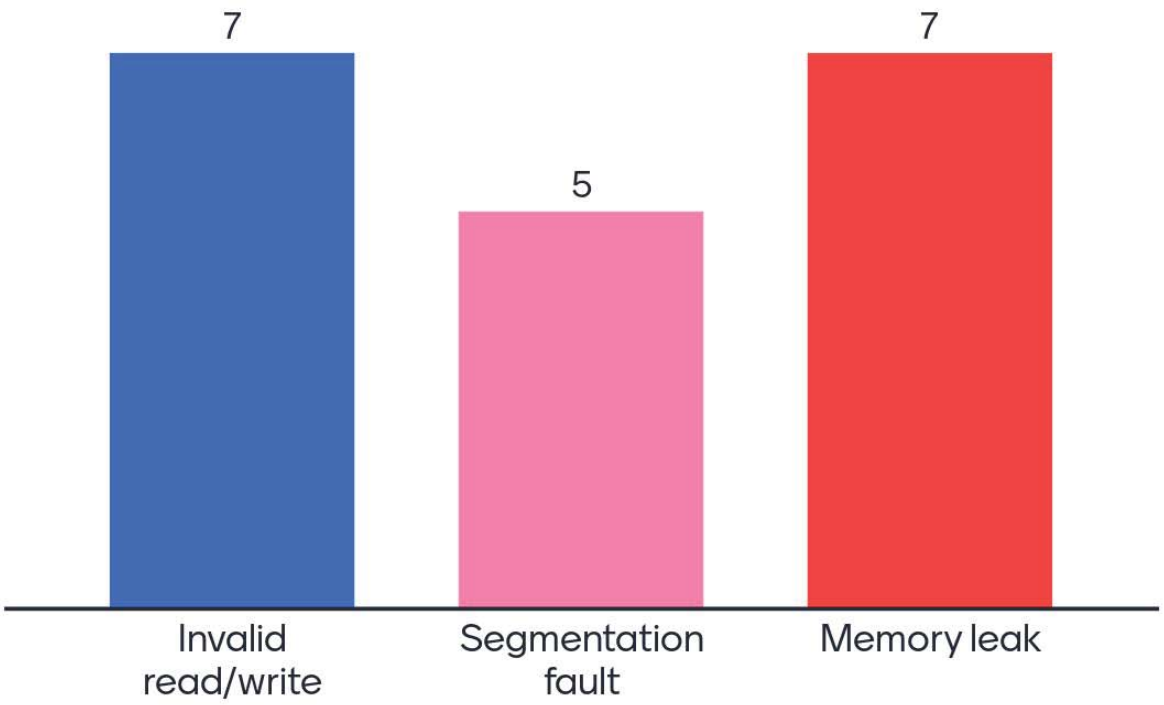
Mentimeter



7

Which of the followings are memory problems that can be detected by valgrind?

Mentimeter



8

What is dynamic memory allocation in C?

Mentimeter

- the heap ✕
- Memory not subject to stack's size limitations and lives as long as the program is running. Use heap memory instead ✕
- The programmer can decide the size of memory needed using malloc and calloc. ✕
- the size of the array can be changed ✕
- storing stuff in heap ✕

The correct answer is: It is a memory management mechanism in C that allows us to allocate memory with dynamic sizes during runtime.

5

What is memory leaks in C?

Mentimeter

- not using free
- happens if you allocate and then don't deallocate
- when we do not use free(). Memory is not closed after use, and cant be used for other things.
- When you forget to deallocate memory

The correct answer is: It happens when we forget to free the memory that we allocated.

6

What is the difference between malloc, realloc, and clalloc?

Mentimeter

Malloc allocates memory to store something. Realloc changes the size of something that has been allocated. calloc like malloc but initializes all to 0

malloc(size) , clalloc init to 0's (num, size) , realloc moves and resizes memory.

malloc set the size of the array, realloc reset the size, and clalloc set the initial to 0

The correct answer is: malloc: new dynamic memory; calloc: same as malloc with values initialized to 0; realloc resize the memory.

3

Quiz

```
1  // Return a C character string containing n exclamation
2  // points. n must be less than 20.
3  char *exclaim(int n) {
4          char s[20];
5          assert(n < 20);
6          for (int i = 0; i < n; i++) {
7                  s[i] = '!';
8          }
9          s[n] = '\0';
10         return s;
11 }
```

What problems can you find in the function shown on the slide?

Mentimeter

the char array s is local. we should dynamically allocate it instead

function name with *

The correct answer is: returning a local variable.

2

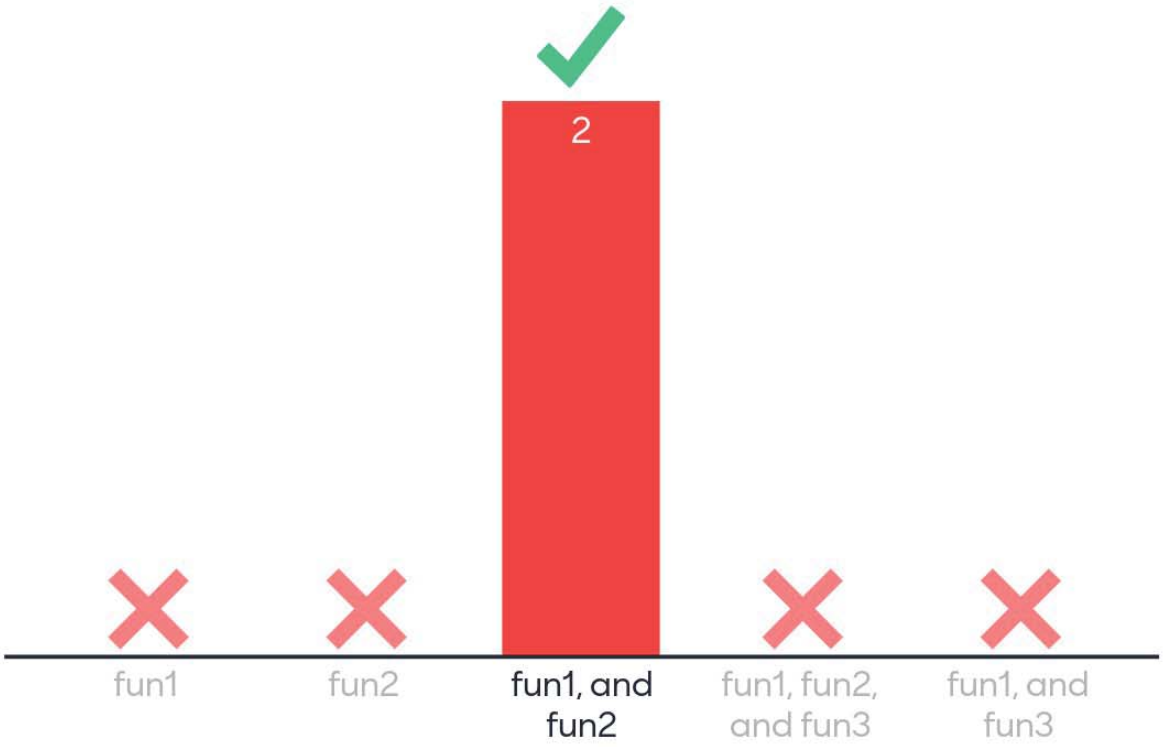
Quiz

Which of the following function(s) has issues with pointers?

```
1  int * fun1(void) {
2          int x= 10;
3          return (&x);
4  }
5  int * fun2(void) {
6          int * px;
7          *px= 10;
8          return px;
9  }
10 int * fun3(void) {
11         int *px;
12         px = (int *) malloc (sizeof(int));
13         *px= 10;
14         return px;
15 }
```

Which of the functions shown on the slide has issues with pointers?

Mentimeter



2

Ask me anything

Mentimeter

0 questions
0 upvotes

Class exercises

Ex 4-2