Lecture 13: String join() & ArrayList

CSE 29: Systems Programming and Software Tools

Olivia Weng

Review: String struct

```
struct string {
    uint64_t length; // = strlen(contents)
    char *contents; // has space for length + null terminator
};
typedef struct string String;
String new_string(char *init_contents);

String plus(String s1, String s2);
```

Demo

- `join()`

How many allocations have been made?

```
// This plus is *just* the heap-allocating version now
String plus(String s1, String s2) {
    uint64_t new_size = s1.length + s2.length + 1;
    char* new_contents = calloc(new_size, sizeof(char));
    strncpy(new_contents, s1.contents, s1.length);
    strncpy(new_contents + s1.length, s2.contents, s2.length);
    new_contents[new_size - 1] = 0;
    String r = { new_size - 1, new_contents };
    return r;
}

String join(String strs[], int count, String delimiter) {
    String s = new_String("");
    for(int i = 0; i < count; i += 1) {
        s = plus(s, strs[i]);
        if(i < count - 1) {
            s = plus(s, delimiter);
        }
    }
    return s;
}

int main() {
    String apple = new_String("apple");
    String banana = new_String("banana");
    String strawberry = new_String("strawberry");
    String fruit[] = { apple, banana, strawberry };

    String comma = new_String(", ");
    String fruitlist = join(fruit, 3, comma);
    printf("%s\n", fruitlist.contents);
}
```

How could we implement an ArrayList in C?

How could we implement a ArrayList class in C?

```
struct list {
    uint32_t size;
    uint32_t capacity;
    String *contents; // the actual list of strings
};
```

What does String *contents look like initially?

```
struct list {
    uint32_t size;
    uint32_t capacity;
    String *contents;
};
```