CSE 333 – SECTION 1

Introduction to and Working with C

Your TAs

- Renshu Gu – PhD Student, EE
- Soumya Vasisht – PhD Student, AA
- Mark Davis – Masters, CSE

- Email are posted on the course website
  - But try to use the staff email instead of our individual emails
- Office hours will be posted soon

- Please use the discussion board!

Questions, Comments, Concerns

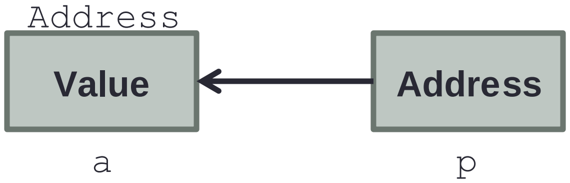
- Do you have any?
- Exercises going ok?
- Lectures make sense?

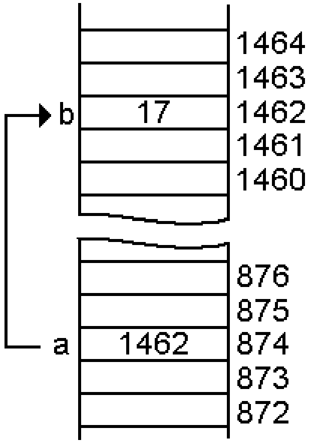
Quick Refresher on C

- General purpose programming language
- Procedural
- Often used in low-level system programming
- Supports use of pointer arithmetic
- Provides facilities for managing memory
- C passes all of its arguments by value
    - Pass-by-reference is simulated by passing the address of a variable

Pointers

- A data type that stores an address
- Used to indirectly refer to values
- Can add/subtract to the address
  - It's just another number

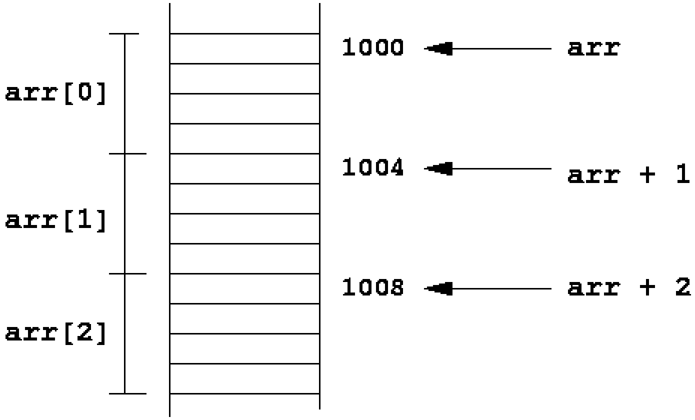




Arrays and pointers

- a[0] <==> *a
- a[3] <==> *(a + 3)

- How about a, a+3,
- *a+3 or *a++?

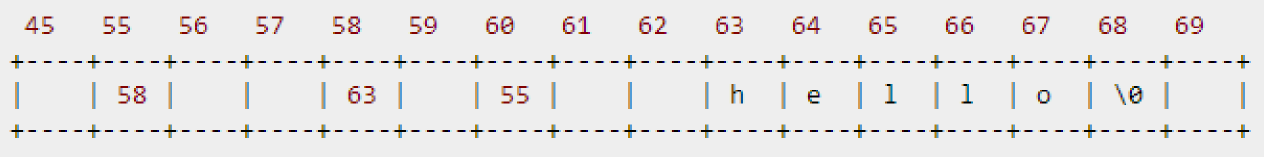


Example

[basic_pointer.c]

```
#include <stdio.h>
void f(int *j) {
  (*j)++;
}
int main() {
  int i = 20;
  int *p = &i;
  f(p);
  printf("i = %d\n", i);
  return 0;
}
```

Pointers to pointers



```
  45   55   56   57   58   59   60   61   62   63   64   65   66   67   68   69
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
|    | 58 |    |    | 63 |    | 55 |    |    | h  | e  | l  | l  | o  | \0 |    |
+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
```

```
char *c = “hello”;
char **cp = &c;
char ***cpp = &cp;
```

- Why could this be useful?

Function pointers

- We can have pointers to functions as well
- Syntax is a little awkward
  - Example: `int (*ptr_to_int_fn)(int, int)`
  - Makes sense if you think about it hard
- We will be using these in the homework assignments!
- Demo: `[function_pointer.c]`

Debugging with gdb

- Just like in CSE 351, gdb is your friend
- Unlike CSE 351, we will be debugging C/C++ code, not assembly
    - Instead of `n`(`ext`)`i` and `s`(`tep`)`i`, use `n`(`ext`) and `s`(`tep`)
- Your first instinct for bug fixing should be gdb, not printf
- Demo: **`[buggy.c]`**

Looking up documentation

- Don’t go straight to Google / Stack Overflow / etc.
- Use the built-in man pages
  - `man <program/utility/function>`
  - `man -f <name>` or `whatis <name>`
  - `apropos <keyword>`
- Much more documentation is linked on the 333 home page
  - Under “Resources” on the left side of the page