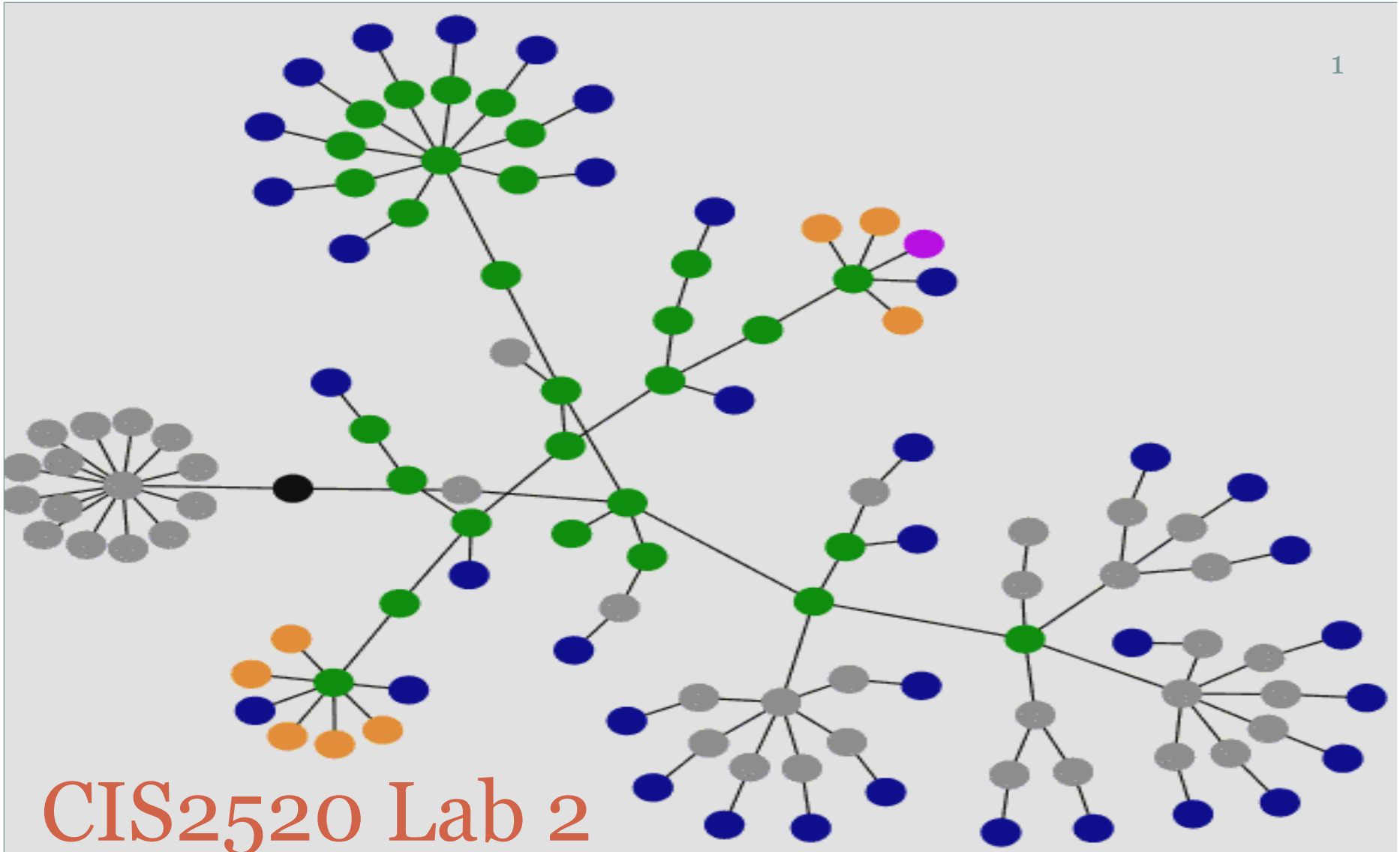


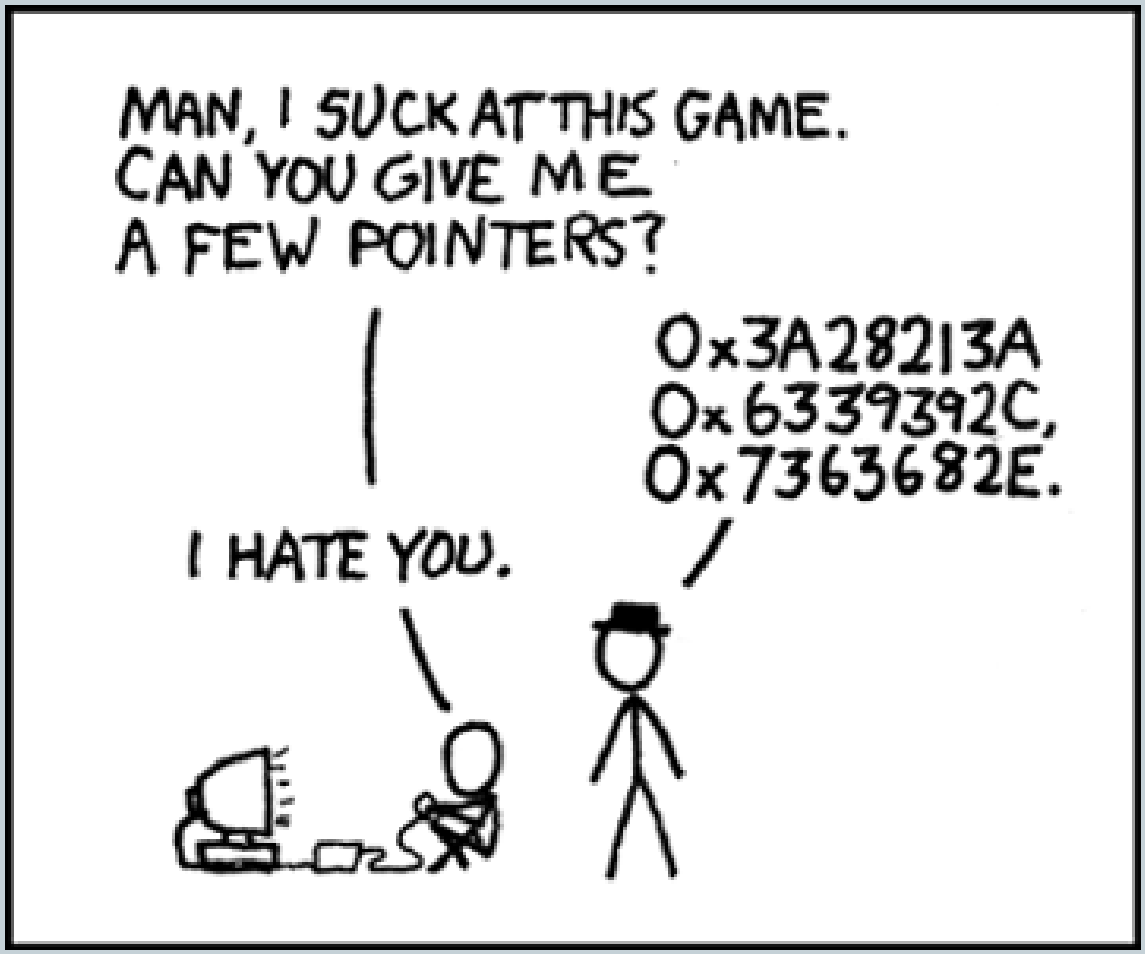
CIS2520 Lab 2

DATA STRUCTURE

Outline

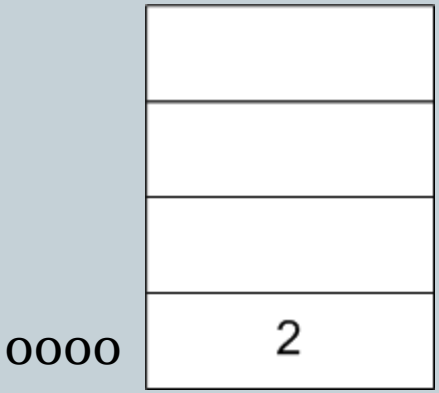
- Pointers review
- Structure and Linked List
- Makefiles review
- Assignment guideline

Pointers


MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A
0x6339392C,
0x7363682E.
I HATE YOU.

Variables in C

- int k = 2;
- A storage type
- A memory address



Pointer variables

- Contain address to a memory location
- Use asterisk to declare a pointer variable

```
int * ptr1;
```

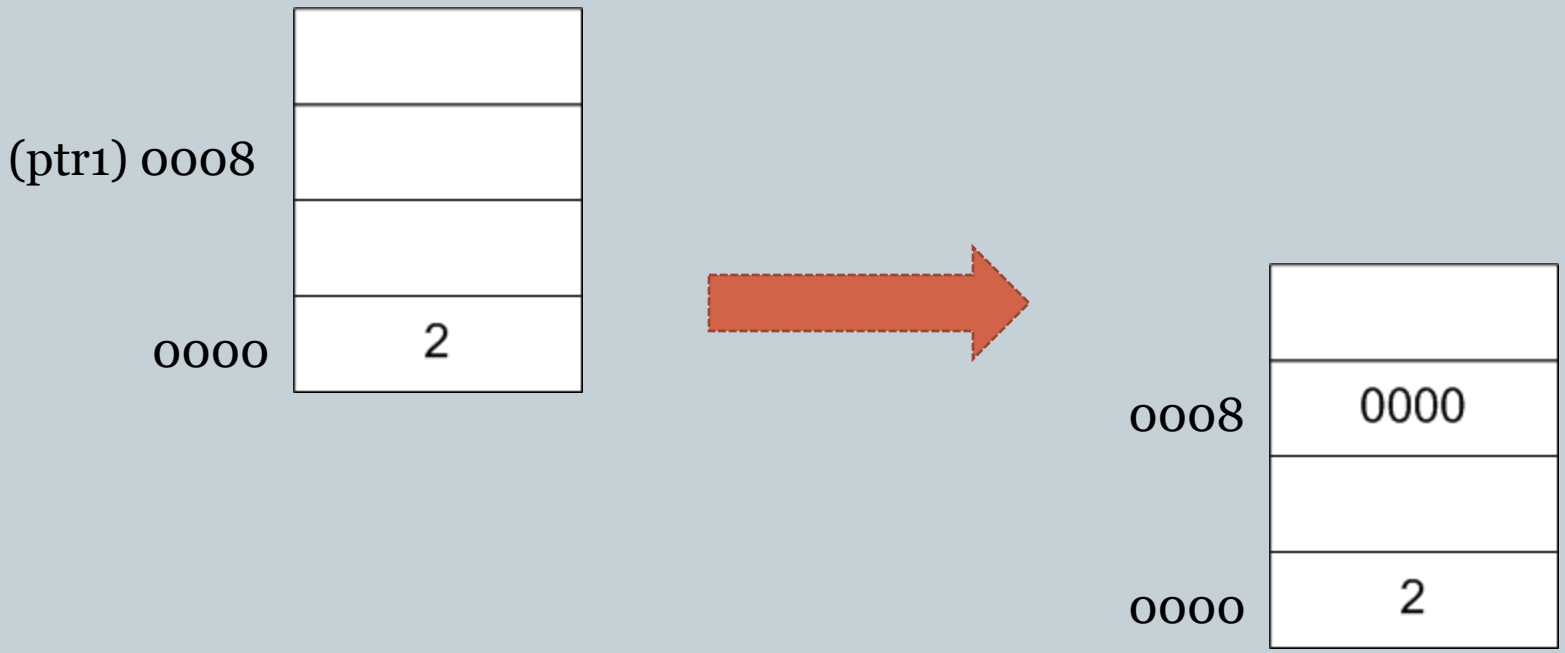
- Use ampersand to reference a variable

```
ptr1 = &k;
```

- Use asterisk to deference a pointer variable

```
*ptr1 = 2;  //assign a value 2 to k
```

Pointer cont’



Pointers cont'

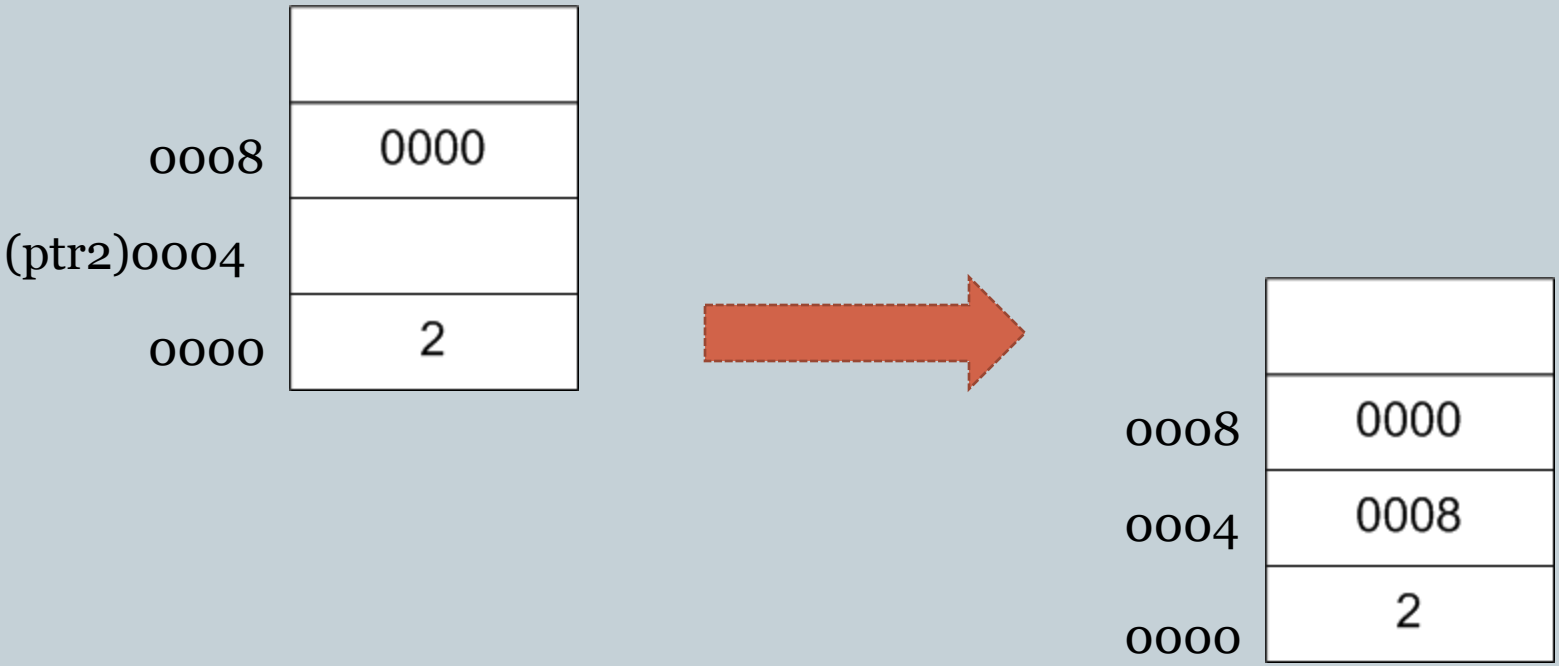
- Pointer of pointers

```
int ** ptr2;
```

- Reference the address of another pointer

```
ptr2 = &ptr1;
```

Pointers cont’

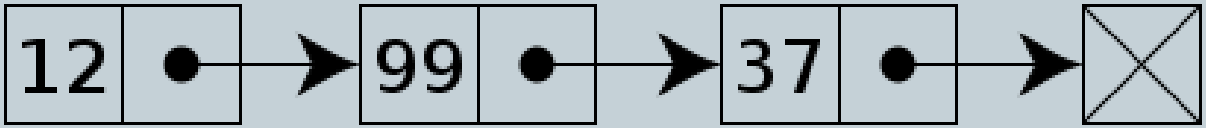


Linked List

- Simple data structure used to implement many other data structures:
  - Stack, queues, hash table, etc..
- Usually implemented using pointers in C

Singly Linked List

- Traverse only in one direction



Singly Linked List cont'


```
typedef struct node{
        int value;
        struct node * next;
}node;
```

```
Node * aNode;
Node * nodePtr;

nodePtr = aNode; //points to beginning of the list

nodePtr = aNode->next; //go to the next node
```

Doubly Linked List

- Traverse in both directions
- Contains two pointers: next and prev

Doubly Linked List cont'


```
typedef struct node{
        int value;
        struct node * next;
        struct node * prev;
}node;

Node * aNode;
Node * nodePtr;

nodePtr = aNode -> next;
nodePtr = aNode -> prev;
```

Makefiles Review

- Target and dependencies:
  - main: sample.o
    - -gcc sample.o –o sample
  - rm:
    - -rm *.o
- Macro
  - CC = -gcc
  - FLAGS = -ansi –Wall –pedantic
  - main: sample.o
    - $(CC) sample.o $(FLAGS) sample

Makefiles cont'

- Example:
  - A program consists of 3 files:
    - main.c //where the main function resides
    - lib1.c
    - lib2.c

Makefiles cont'


```
CC=gcc
FLAGS=-ansi –Wall
OBJECTS=main.o lib1.o lib2.o

main: $(OBJECTS)
        -$(CC) $(OBJECTS) $(FLAGS) –o main
main.o: main.c
        -$(CC) –c main.c
lib1.o: lib1.c
        -$(CC) –c lib1.c
lib2.o: lib2.c
        -$(CC) –c lib2.c
rm:
        -rm *.o
```

Assignment Guidelines

- Requirements:
  - Makefile
  - Tar and gzip of files
  - Comments appropriately
  - Name and date at the beginning of each file
  - Indentation
  - Submit via the moodle website

Suggested Readings:

- Makefiles:
  - www.gnu.org/software/make/manual/make.html
- Thanks:
  - Wikipedia - images