CS 161
Intro to CS I

Stack vs. Heap and 1-d Arrays
In-class Exercise

Pointers vs. References

• What if you made a pointer (p2) that points to a pointer (p) to an int (x)?
  – What would the picture look like?
  – Write the code for this picture.

• Can you make this same picture for references? **No**
  – What if you had two references, r and r2?
```cpp
#include <iostream>
#include <string>

using namespace std;

void fun(string *str){
    *str="jen"; //dereference to get to string s in main
    cout << (*str).at(0) << endl;
    cout << str->at(0) << endl; //convention
}

int main() {
    string s="hi";
    cout << s << endl;
    fun(&s); //address of adds a * to type, i.e. &s is a string *
    cout << s << endl;
    return 0;
}
```
Revisit Variables vs. Pointers

• Value Semantics
  – Values stored directly
  – Copy of value is passed
  ```c
  int i, j=2;
i=j;
  ```

• Pointer Semantics
  – Address to variable is stored
  – Copy of address is passed
  ```c
  int *i, j=2;
i=&j;
  ```
What if we don’t have the j?

• We need to create the address space.
• How do we do this?
  – new type;
• For example:
  ```cpp
  int *i;
  i = new int;  //new returns an address
  *i = 10;
  ```
Binky Pointer Video

• Watch the C++ Stanford Binky video:
  http://cslibrary.stanford.edu/104/

... and make sure you don’t blow binky’s head off in the future😊
Stack vs. Heap

• Static vs. Dynamic
Static vs. Dynamic

- **Static Semantics**
  - Assign address of variable
    ```
    int *i, j=2;
    i=&j;
    ```

- **Dynamic Semantics**
  - Create memory
  - Assign memory to pointer
    ```
    int *i=NULL;
    i=new int;
    *i=2;
    ```
What About Memory Leaks?

• What happens here...

...  
int main () {  
    int *i=NULL; //created in main function  
    while(1) {  
        i = new int;  
    }  
}
Fixing Memory Leaks...

• What happens here...

... int main () {
  int *i=NULL; //created in main function
  while(1) {
    i = new int;
    delete i; //free memory that i points to, preventing mem leaks
  }
}

OSU Oregon State University
#include <iostream>
#include <string>
using namespace std;

void fun(string *str){
    *str="jen"; //dereference to get to string s in main
    cout << (*str).at(0) << endl;
    cout << str->at(0) << endl; //convention
    while(1) {
        str=new string; //create new string on heap
        *str="hello"; //dereference to get to string on heap
        cout << str->at(0) << endl;
        cout << &str << endl; //address of str, which is on stack
        cout << str << endl; //address of string on heap
        delete str; //delete memory str points to, otherwise memory leak
    }
}

int main() {
    string s="hi";
    cout << s << endl;
    fun(&s); //address of adds a * to type, i.e. &s is a string *
    cout << s << endl;
    return 0;
}
What is an Array?

- **Array (ar·ray) n.** An ordered arrangement of related items.
  - Example: Array of colors in a rainbow.
    - Related items?
    - Ordered arrangement?
  - Class examples?
  - Computer Science
    - Same data type/data structure
    - Contiguous memory locations
Create 1-D Array

- How do you access each item?
- What does the array name represent?
- Why is the array name the address of 1st element?
- What are the initial values?