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.

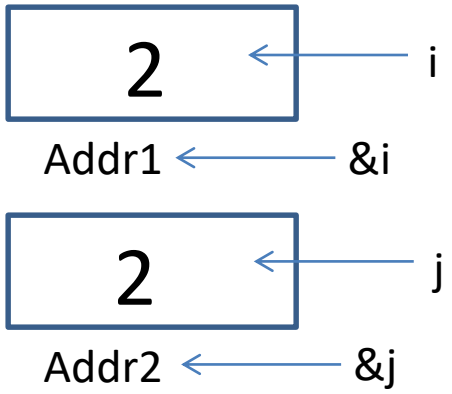
- Can you make this same picture for references?
  - What if you had two references, r and r2?

String Pointers Demo

Revisit Variables vs. Pointers

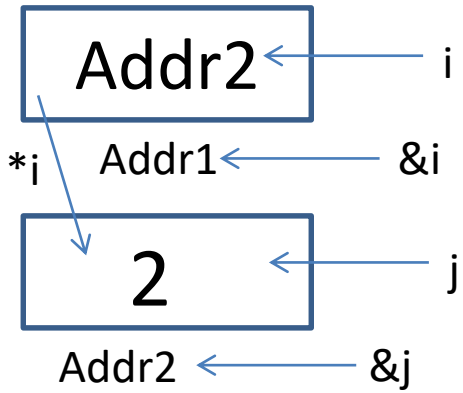
- Value Semantics
  - Values stored directly
  - Copy of value is passed

```
int i, j=2;
i=j;
```



- Pointer Semantics
  - Address to variable is stored
  - Copy of address is passed

```
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:

```
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

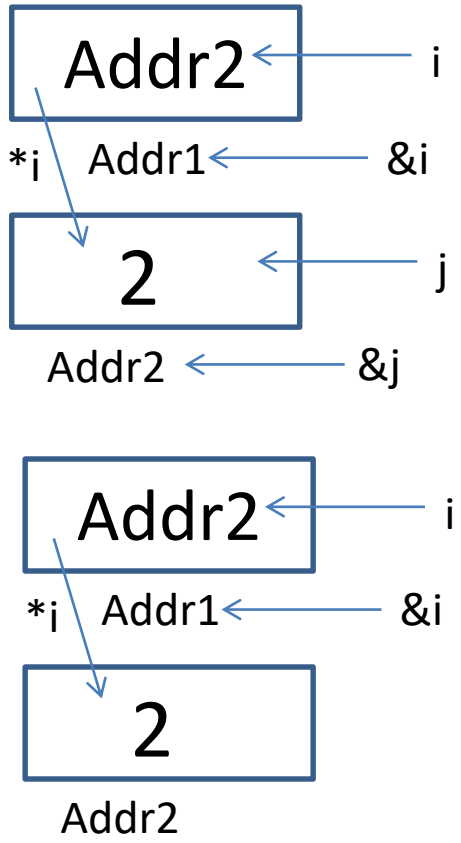
  int *i, j=2;

  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
  }
}
```

Dynamic Memory Demo…

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

int student_grades[5];

- 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?

Initialize/Assign Values

- **Declaration**

  int student_grades[5] = {0, 0, 0, 0, 0};

- **Individual Elements**

  student_grades[0]=0;

  …

  student_grades[4]=0;

- Why is this incorrect?

  student_grades={0, 0, 0, 0, 0};

Initialize/Assign Values…

- **Using a Loop**

  While Loop Example:

```
i=0;
while (i<5)  {
   student_grades[i]=0;
   i++;
}
```

  For Loop Example:

```
for(i=0; i<5; i++)
   student_grades[i]=0;
```

- Which is better to use with arrays and why?

Read/Print 1-D Array Values

- Read Values From User

```
for(i=0; i<5; i++)  {
    cout << "Enter final grade for student: ";
    cin >> student_grades[i];
}
```

- Print Values

```
for (i=0; i<5; i++)  {
    cout << "Student\'s final grade is " << student_grades[i] << endl;
}
```

Static vs. Dynamic 1-D arrays…