C++ Interlude 2
Pointers and Memory allocation

APPLICATION MEMORY

Function Parameters and Local Variables

Memory is freed when function ends

Objects instantiated with `new`

Memory is freed when `delete` statement is executed

Application Memory

Activation Record
Activation Record
Activation Record

Application Stack

Activation Record

Free Store
(Heap)

Static Storage

Application Code

Pearson

APPLICATION MEMORY

```cpp
#include <iostream>
#include <string>
#include "PlainBox.h"
using namespace std;


PlainBox<string>* makePlainBox(string something)
{
    PlainBox<string>* myBox = new PlainBox<string>(something);


    return myBox;
}


int main()
{
    string gift = "Ring";
    PlainBox<string>* giftBox = makePlainBox(gift);
    cout << giftBox->getItem() << endl;
    delete giftBox;
    giftBox = nullptr;
    return 0;
}
```

Application Memory

Activation Record   `main`
`gift`   `Ring`
`giftBox`   `nullptr`

Activation Record   `makePlainBox`
`something`   `Ring`
`myBox`

Dangling Pointer

PlainBox Object
`item`

Static Storage

Application Code

Pearson

APPLICATION MEMORY

```cpp
#include <iostream>
#include <string>
#include "PlainBox.h"
using namespace std;

PlainBox<string>* makePlainBox(string something)
{
    PlainBox<string>* myBox = new PlainBox<string>(something);
    return myBox;
}

int main()
{
    PlainBox<string>* giftBox = new PlainBox<string>();  nullptr;
    string gift = "Ring";
    giftBox = makePlainBox(gift);
    cout << giftBox->getItem() << endl;
    delete giftBox;
    giftBox = nullptr;
    return 0;
}
```

Memory Leak

Application Memory

Activation Record   `main`
`gift` `Ring`
`giftBox` `nullptr`

PlainBox Object
`item` `Ring`

PlainBox Object
`item` `unknown`

Static Storage

Application Code

Pearson