Heap Memory

Prof. Wade Fagen-Ulmschneider

Heap memory allows us to create memory independent of the lifecycle of a function.

Heap Memory

If memory needs to exist for longer than the lifecycle of the function, we must use **heap memory**.

- The only way to create heap memory in C++ is with the **new** operator.

The **new** operator returns a **pointer** to the memory storing the data – not an instance of the data itself.

C++'s New Operator

The `new` operator in C++ will always do three things:

1. Allocate memory on the heap for the data structure
2. Initialize the data structure
3. Return a pointer to the start of the data structure

The memory is only ever reclaimed by the system when the pointer is passed to the `delete` operator.

Heap Memory

The code below allocates two chunks of memory:

- Memory to store an **integer pointer** on the **stack**
- Memory to store an **integer** on the **heap**

```
int * numPtr = new int;
```

cpp-heapMemory/main.cpp

```cpp
int main() {
  int *numPtr = new int;

  std::cout << "*numPtr: " << *numPtr << std::endl;
  std::cout << " numPtr: " << numPtr << std::endl;
  std::cout << "&numPtr: " << &numPtr << std::endl;

  *numPtr = 42;
  std::cout << "*numPtr assigned 42." << std::endl;

  std::cout << "*numPtr: " << *numPtr << std::endl;
  std::cout << " numPtr: " << numPtr << std::endl;
  std::cout << "&numPtr: " << &numPtr << std::endl;

  return 0;
}
```

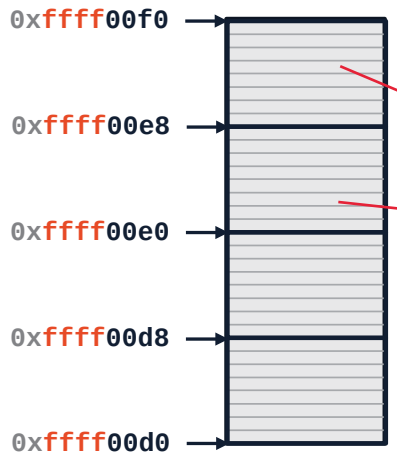
Address of "new int" in heap memory

Address of pointer "numPtr" in stack memory

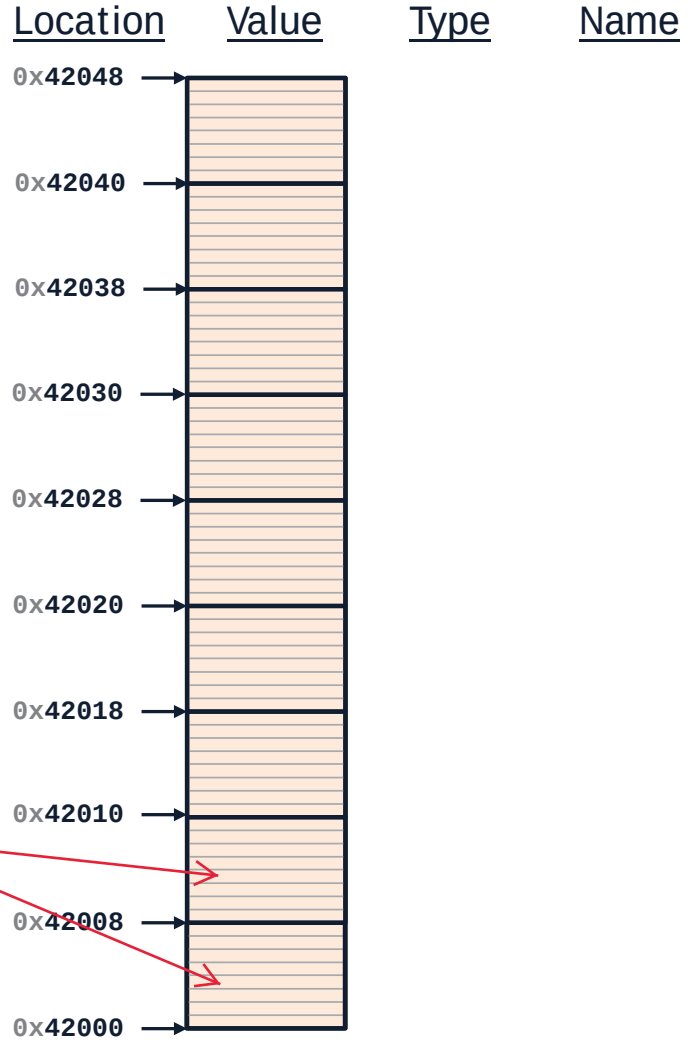
cpp-heapMemory/heap1.cpp

```
11 int main() {
12   int *p = new int;
13   Cube *c = new Cube;
14
15   *p = 42;
16   (*c).setLength(4);
17
18   delete c;
19   delete p;
20   return 0;
21 }
```

only delete heap memory



Stack Memory



Heap Memory

nullptr

The C++ keyword `nullptr` is a pointer that points to the memory address 0x**0**.

- **nullptr** represents a pointer to “nowhere”
- Address 0x0 is reserved and never used by the system.
- Address 0x0 will always generate an “segmentation fault” when accessed.
- Calls to `delete` 0x0 are ignored.

Arrow Operator (->)

When an object is stored via a pointer, access can be made to member functions using the `->` operator:

```
c->getVolume();
```

…identical to…

```
(*c).getVolume();
```

cpp-heapMemory/heap2.cpp

```
11  int main() {
12    Cube *c1 = new Cube;
13    Cube *c2 = c1;
14
15    c2->setLength( 10 );
16
17    delete c2;
18    delete c1; // !!
19
20    return 0;
21  }
```

Memory is deleted by c2, so "delete c1" cause problem.