CS 225

#3: Memory

September 1, 2017

Sphere – Leveled UP:

sphere.h

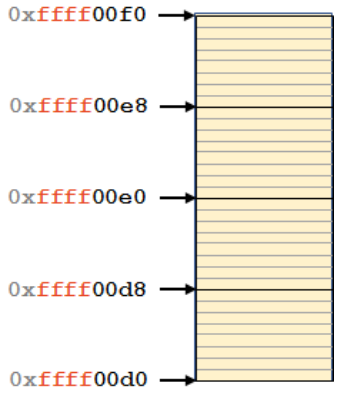
```
1  #ifndef SPHERE_H
2  #define SPHERE_H
3  namespace cs225 {
4    class Sphere {
5    public:
6      Sphere();
7      Sphere(double r);
8      double getRadius();
9      double getVolume();
10
11
12
13
14   private:
15     double r_;
16
17   };
18 }
19
20 #endif
```

sphere.cpp

```
1  #include "sphere.h"
2  namespace cs225 {
3   Sphere::Sphere()
    {
4
5   }
6
7   Sphere::Sphere(double r){
8     r_ = r;
9     // lots of other logic…
10  }
11
12  double Sphere::getRadius(){
13    return r_;
14  }
15
16  double Sphere::getVolume(){
17    return (4 * r_ * r_ *
18        r_ * 3.14159) / 3.0;
19  }
20 }
```

Idea: _____ allow us to invoke other constructors to increase code reuse and eliminate copy/pasted code.

Stack Memory



example1.cpp

```
1  int main() {
2    int a;
3    int b = -3;
4    int c = 12345;
5
6    int *p = &b;
7
8    return 0;
9  }
```

Location	Value	Type	Name
0xffff00f0 →			
0xffff00e8 →			
0xffff00e0 →			
0xffff00d8 →			
0xffff00d0 →			

example2.cpp

```
3  int main() {
4    cs225::Sphere s;
5    cs225::Sphere *p = &s;
6
7    return 0;
8  }
```

Location	Value	Type	Name
0xffff00f0 →			
0xffff00e8 →			
0xffff00e0 →			
0xffff00d8 →			
0xffff00d0 →			

Stack Frames

All variables (including parameters to the function) that are part of a function are part of that function's **stack frame**. A stack frame:

1.

2.

stackframe.cpp

```
1  int hello() {
2    int a = 100;
3    return a;
4  }
5
```

```
6  int main() {
7    int a;
8    int b = -3;
9    int c = hello();
10   int d = 42;
11
12   return 0;
13 }
```

Location	Value	Type	Name
0xffff00f0 →			
0xffff00e8 →			
0xffff00e0 →			
0xffff00d8 →			
0xffff00d0 →			

INSIGHT PUZZLE – What happens here?

sphere-badCreate.cpp

```
4  Sphere *CreateUnitSphere() {
5    Sphere s(1);
6    return &s;
7  }
8
9  int main() {
10   Sphere *s = CreateUnitSphere();
11   double r = s->getRadius();
12   double v = s->getVolume();
13   return 0;
14 }
```

Location	Value	Type	Name
0xffff00f0 →			
0xffff00e8 →			
0xffff00e0 →			
0xffff00d8 →			
0xffff00d0 →			

Big Idea: Heap Memory

heap1.cpp

```
4  int main() {
5    int *p = new int;
6    int *s = new Sphere(10);
7
8    return 0;
9  }
```

Stack	Value
0xffff00f0 →	
0xffff00e8 →	
0xffff00e0 →	
0xffff00d8 →	
0xffff00d0 →	

Heap	Value
0x42020 →	
0x42018 →	
0x42010 →	
0x42008 →	
0x42000 →	

heap2.cpp

```
4  int main() {
5    Sphere *s1 = new Sphere();
6    Sphere *s2 = s1;
7
8    s2->setRadius( 10 );
9
10   return 0;
11 }
```

Stack	Value
0xffff00f0 →	
0xffff00e8 →	
0xffff00e0 →	
0xffff00d8 →	
0xffff00d0 →	

Heap	Value
0x42020 →	
0x42018 →	
0x42010 →	
0x42008 →	
0x42000 →	