Name:

Due: Monday, Mar 19th at noon.

Assumptions: For all of these problems you may assume the following

- sizeof(int) = 4; sizeof(short) = 2; sizeof(char) = 1; all pointers require 4 bytes
- The stack starts at address 100 and grows up.
- The heap starts at address 400 and grows up.
- ?? represents an unknown/uninitialized value.

Given the following code:

Question 1 (32 points):

int x = 3; /* x is at address 100 */
int y = 4; /* y is at address 104 */
int & r = x; /* r is at address 108 */
int * p = &r; /* p is at address 112 */

/* Location 1 */
r = 6;
/* Location 2 */
r = y;
y = 11;
/* Location 3 */
*p = 12;
/* Location 4 */

Fill in the following table with the values of x, y, r, and p at the above indicated 4 locations:

<table>
<thead>
<tr>
<th>Loc 1</th>
<th>Loc 2</th>
<th>Loc 3</th>
<th>Loc 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>r</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>p</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Question 2 (36 points): Class Rectangle has fields length and width, as well as virtual methods area and perimiter. Class Material has fields thickness and hardness as well as virtual method smash. Class Surface extends Rectangle (which is its primary parent) and Material. The object layout for Rectangle and Material are as follows:
Question 3 (22 points): Given the object layout of Material, Rectangle, and Surface objects from question 2. Suppose that in the following code, the Surface object is allocated at address 400 (i.e. s = 400)

    Surface * s = new Surface();
Rectangle * r = s;
Material * m = s;

1. What is the numerical value of r (i.e. what address does it point to)?
2. What is the numerical value of m?
3. Does the expression r == s evaluate to true (non-zero) or false (zero)?
4. Does the expression m == s evaluate to true (non-zero) or false (zero)?
5. Suppose that Surface overrides all of the methods it inherits from its parent class. Give an example of a method invocation on one of s, r, or m which has additional overhead due to multiple inheritance. Explain what this overhead is, and why it is needed.

---

**Question 4 (10 points):** Consider the following class hierarchy:

```cpp
class A { ... };
class B : virtual public A { ... };
class C : virtual public A { ... };
class D : public B, public C { ... };
```

1. What is virtual inheritance (i.e. what is different about this inheritance than if
virtual did not appear in the declaration of classes B and/or C?

2. Why must virtual be specified in the declarations of B and C, as opposed to in the declaration of the class where multiple inheritance occurs (i.e. something like class D: public B, public C (virtual A) { ... } ;