368 Worksheet: Pointers and Structs

```
struct Coord { float x; float y; };
struct City  { int population; };
struct House {
  int beds;
  int baths;
  Coord coord;
  City* city;
};

City A{.population=1000};
City B{.population=1000};
House house =
      {.beds=3, .baths=2,
      .coord={.x=-89, .y=43},
      .city=&A};
House* p = &house;
House** p2 = &p;
```

addr	memory	notes
32	1000	City A
36	1000	City B
40	3	House house
44	2	
48	-89	
52	43	
56	32	
60		
64	40	House* p
68		
72	64	House** p2
76		

1. rewrite p->beds, without using the "->" operator: _____
2. complete to get the x coord (-89): `house____coord____x`
3. complete to get population (1000): `house___city___population`
4. complete to get the y coord (-89): `p____coord____y`
5. complete to get population (1000): `p____city____population`
6. starting from p2, get the number of baths (2): _____
7. is (house.city->population == B.population) true or false? _____
8. is (house.city == &B) true or false? _____

368 Worksheet: new and delete

Add delete and delete[] calls as necessary to achieve correct and leak-free code.

```
int *mult(int *arr, int count, int factor) {
  int* result = new int[count];
  for(int i=0; i<count; i++)
    result[i] = arr[i] * factor;


  return result;
}

void mult_sum(int *arr, int count,
              int factor, int* result) {
  auto arr2 = mult(arr, count, factor);
  for(int i=0; i<count; i++) {
    *result += arr2[i];
  }


}

void test() {
  int a[3] = {1,2,3};
  int x = 0;
  int* y = &x;
  mult_sum(a, sizeof(a)/sizeof(a[0]), 2, y);
  cout << x << "\n";


  int* z = new int{10};
  mult_sum(a, sizeof(a)/sizeof(a[0]), -1, z);
  cout << *z << "\n";


}
```