Questions:

1. Declare a pointer to a float and a pointer to a char.
2. What do we use the *sizeof()* operator for? If we have two integers, *ax* and *bx*, defined one after the other, how can we simulate the purpose of the *sizeof()* operator?
3. Explain, in words, what the following code snippet does. Make sure to use the words “dereference”, “pointer” and “reference (&)” inside your explanation.

double num1 = 7.0;

double& num2 = num1;

double\* dptr = &num1;

cout << num1 << endl;

cout << &num1 << endl;

num2 = 33.4;

\*dptr = 1234.567;

cout << num2 << endl;

cout << &num2 << endl;

1. Suppose the operating system you were working on defines a pointer to an int as 8 bytes (64 bits) wide. How wide would a pointer to a short be? A string pointer? Explain.
2. Explain, using your knowledge of the relationship between pointers and arrays, on why the following code produces its output.

|  |  |
| --- | --- |
| **Syntax** | **Output** |
| #include <iostream>  using namespace std;  void printArrayMultByTwo(int \* int\_list, const int SIZE)  {  for(int i = 0; i < SIZE; ++i)  {  int\_list[i] \*= 2;  cout << int\_list[i] << " ";  }  cout << endl << endl;  }  int main()  {  const int LIST\_SIZE = 5;  int my\_list[] = {27, 32, 55, 1, 3752};  printArrayMultByTwo(my\_list, LIST\_SIZE);  printArrayMultByTwo(my\_list, LIST\_SIZE);  printArrayMultByTwo(my\_list, LIST\_SIZE);    return 0;  } | 54 64 110 2 7504  108 128 220 4 15008  216 256 440 8 30016 |

1. Using the code on the left, make a main function that creates a pointer that dynamically creates a Rectangle, sets its width to 3.0 and height to 4.0, prints out the rectangle, and deletes the Rectangle afterwards. Assume all the methods are well-defined.

|  |  |
| --- | --- |
| **Given Code** | **Your Main Function** |
| class Rectangle  {  double width, height;  public:  Rectangle(); // defaults width = height = 1.0  Rectangle(double w, double h);    // getters  double getWidth();  double getHeight();    // setters  void setWidth(double w);  void setHeight(double h);  // miscellaneous  double getArea();  void printSelf();  }; |  |