Even numbers from 1 to 36 on pages 697-699.

2. The **&** operator can be used to determine a variable’s address.

4. The **indirection \*** operator can be used to work with the variable a pointer points to.

6. Creating variables while a program is running is called **dynamic memory allocation.**

8. If the new operator cannot allocate the amount of memory requested, it throws **bad\_alloc.**

10. When a program is finished with a chunk of dynamically allocated memory, it should free it with the **delete** operator.

12. What does the indirection operator do? **dereferences a pointer**

14. Name two different used for the C++ operator \*. **Multiplication, declaring a pointer**

16. Assuming that ptr is a pointer to an int, what happens when you add 4 to it?

**The pointer will point to a new address that’s four places ahead in memory**

18. What is the purpose of the new operator?

**To dynamically allocate memory**

20. Under what circumstances can you successfully return a pointer from a function?

**When the variable being pointed to is not local to the function**

22. What is the difference between a pointer to a constant and a constant pointer?

**Pointer to a constant: the variable pointed to can’t be change, but the pointer itself can be changed.**

**Constant Pointer: the pointer can’t be changed, but the variable being pointed to can**

24. Show C++ code for defining a variable ptr that is a constant pointer to int.

**int \* const ptr;**

26. Name the header file that needs to be included in a program that uses smart pointers.

**#include <memory>**

28. What does the get() method of the unique\_ptr class do?

**Returns the stored pointer that points to the object managed by the unique\_ptr**

30. List three different operations that are permitted on raw pointers but not on unique\_ptr objects.

**unique\_ptr cannot be copied**

32. What problems are likely to occur if you have the following declaration in your program?

shared\_ptr<double []> sDPtr;

**Its destructor will run and destroy all of the double object elements**

34. Consider the function:

void modify(int & x)

{

x = 10

}

Show how to call the modify function so that it sets the integer int i; to 10.

**modify(i);**

36. Write a function

void switchEnds(int \*array, int size);

that is passed the address of the beginning of an array and the size of the array. The function swaps the values in the first and last entries of the array.

**void switchEnds(int \*array, int size)**

**{**

**int temp = \*(array);  
(array) = (array+size-1);  
\*(array+size-1) = temp;**

**}**