Name: CAELAN KIMBALL **ID: 020022**
Date: 03/02/2020 **Class: Computer Programming 2**
Score: 35 / 40 (87.5%)

Comp. Prog. Chapter 9 Test

TRUE/FALSE

F 1. True/False: With pointer variables you can access, but you cannot modify, data in other variables.

Points: 1 / 1

F 2. True/False: An array name is a pointer constant because the address stored in it cannot be changed during runtime.

Points: 0 / 1

T 3. True/False: It is legal to subtract a pointer variable from another pointer variable.

Points: 1 / 1

T 4. True/False: A pointer can be used as a function argument, giving the function access to the original argument.

Points: 1 / 1

F 5. True/False: The ampersand (`&`) is used to dereference a pointer variable in C++.

Points: 1 / 1

T 6. True/False: Assuming `myValues` is an array of `int` values, and `index` is an `int` variable, both of the following statements do the same thing.

```
cout << myValues[index] << endl;

cout << *(myValues + index) << endl;
```

Points: 1 / 1

T 7. True/False: In C++ 11, you can use smart pointers to dynamically allocate memory and not worry about deleting the memory when you are finished using it.

Points: 1 / 1


T 8. True/False: To use any of the smart pointers in C++ 11, you must `#include` the `memory` header file with the following directive:

```
#include <memory>
```

Points: 1 / 1

MULTIPLE CHOICE


B 9. The _____, also known as the address operator, returns the memory address of a variable.

a. asterisk (*)
b. ampersand (&)
c. percent sign (%)
d. exclamation point (!)
e. None of these

Points: 1 / 1


C 10. With pointer variables, you can _____ manipulate data stored in other variables.

a. never
b. seldom
c. indirectly
d. All of these
e. None of these

Points: 1 / 1


D 11. The statement:

```
int *ptr = nullptr;
```

has the same meaning as _____.

a. `int ptr = nullptr;`
b. `*int ptr = nullptr;`
c. `int ptr* = nullptr;`
d. `int* ptr = nullptr;`
e. None of these

Points: 1 / 1

C 12. When you work with a dereferenced pointer, you are actually working with _____.

a. a variable whose memory has been allocated
b. a copy of the value pointed to by the pointer variable
c. the actual value of the variable whose address is stored in the pointer variable
d. All of these
e. None of these

Points: 1 / 1

A 13. _____ can be used as pointers.

a. Array names
b. Numeric constants
c. Punctuation marks
d. All of these
e. None of these

Points: 1 / 1

C 14. The contents of pointer variables may be changed with mathematical statements that perform _____.

a. all mathematical operations that are legal in C++
b. multiplication and division
c. addition and subtraction
d. B and C
e. None of these

Points: 1 / 1

A 15. In C++ 11, the _____ key word was introduced to represent the address 0.

a. `nullptr`
b. `NULL`
c. `weak_ptr`
d. All of these
e. None of these

Points: 1 / 1

 D 16. What does the following statement do?

```
double *num2;
```

a. Declares a `double` variable named `num2`.
b. Declares and initializes an pointer variable named `num2`.
c. Initializes a variable named `*num2`.
d. Declares a pointer variable named `num2`.
e. None of these

Points: 1 / 1

 C 17. When the less than (<) operator is used between two pointer variables, the expression is testing whether _____.

a. the value pointed to by the first is less than the value pointed to by the second
b. the value pointed to by the first is greater than the value pointed to by the second
c. the address of the first variable comes before the address of the second variable in the computer's memory
d. the first variable was declared before the second variable
e. None of these

Points: 1 / 1

 C 18. Look at the following statement:

```
sum += *array++;
```

This statement _____.

a. is illegal in C++
b. will always result in a compiler error
c. assigns the dereferenced pointer's value, then increments the pointer's address
d. increments the dereferenced pointer's value by one, then assigns that value
e. None of these

Points: 1 / 1

C 19. Use the `delete` operator only on pointers that were _____.

a. never used
b. not correctly initialized
c. created with the `new` operator
d. dereferenced inappropriately
e. None of these

Points: 1 / 1

A 20. A function may return a pointer, but the programmer must ensure that the pointer _____.

a. still points to a valid object after the function ends
b. has not been assigned an address
c. was received as a parameter by the function
d. has not previously been returned by another function
e. None of these

Points: 1 / 1

C 21. Which of the following statements is not valid C++ code?

a. `int ptr = &num1;`
b. `int ptr = int *num1;`
c. `float num1 = &ptr2;`
d. All of these are valid.
e. All of these are invalid.

Points: 0 / 1

C 22. Which of the following statements deletes memory that has been dynamically allocated for an array?

a. `int array = delete memory;`
b. `int delete[ ];`
c. `delete [] array;`
d. `new array = delete;`
e. None of these

Points: 1 / 1

C 23. When this is placed in front of a variable name, it returns the address of that variable.

a. asterisk (`*`)
b. conditional operator
c. ampersand (`&`)
d. semicolon (`;`)
e. None of these

Points: 1 / 1

B 24. What will the following statement output?

```
cout << &num1;
```

a. The value stored in the variable called `num1`
b. The memory address of the variable called `num1`
c. The number 1
d. The string "&num1"
e. None of these

Points: 1 / 1

C 25. Look at the following statement:

```
int *ptr = nullptr;
```

In this statement, what does the word `int` mean?
a. The variable named `*ptr` will store an integer value.
b. The variable named `*ptr` will store an asterisk and an integer value.
c. `ptr` is a pointer variable that will store the address of an integer variable.
d. All of these
e. None of these

Points: 1 / 1

C 26. The _____ and _____ operators can be used to increment or decrement a pointer variable.
a. addition, subtraction
b. modulus, division
c. ++, --
d. All of these
e. None of these

Points: 1 / 1

A 27. Not all arithmetic operations may be performed on pointers. For example, you cannot _____ or _____ a pointer.
a. multiply, divide
b. add, subtract
c. +=, -=
d. increment, decrement
e. None of these

Points: 1 / 1

D 28. Which statement displays the address of the variable `num1`?

a. `cout << num1;`
b. `cout << *num1;`
c. `cin >> &num1;`
d. `cout << &num1;`
e. None of these

Points: 1 / 1

B 29. Dynamic memory allocation occurs _____.

a. when a new variable is created by the compiler
b. when a new variable is created at runtime
c. when a pointer fails to dereference the right variable
d. when a pointer is assigned an incorrect address
e. None of these

Points: 1 / 1

C 30. The following statement:

```
int *ptr = new int;
```

a. results in a compiler error
b. assigns an integer less than 32767 to the variable named `ptr`
c. assigns an address to the variable named `ptr`
d. creates a new pointer named `int`
e. None of these

Points: 1 / 1

D 31. If you are using an older compiler that does not support the C++ 11 standard, you should initialize pointers with _____.

a. the integer 0, or the value `NULL`
b. the null terminator `'\0'`
c. a nonzero value
d. All of these
e. None of these

Points: 0 / 1

B 32. Every byte in the computer's memory is assigned a unique _____.

a. pointer
b. address
c. dynamic allocation
d. name
e. None of these

Points: 1 / 1

B 33. When you pass a pointer as an argument to a function, you must _____.

a. declare the pointer variable again in the function call
b. dereference the pointer variable in the function prototype
c. use the `#include<func_ptr.h>` statement
d. not dereference the pointer in the function's body
e. None of these

Points: 0 / 1

B 34. A pointer variable may be initialized with _____.

a. any non-zero integer value
b. a valid address in the computer's memory
c. an address less than 0
d. A and C only
e. None of these

Points: 1 / 1

C 35. If a variable uses more than one byte of memory, for pointer purposes its address is _____.

a. the address of the last byte of storage
b. the average of the addresses used to store the variable
c. the address of the first byte of storage
d. general delivery
e. None of these

Points: 1 / 1

B 36. What will the following code output?

```
int number = 22;
int *var = &number;
cout << *var << endl;
```

a. The address of the `number` variable
b. 22
c. An asterisk followed by 22
d. An asterisk followed by the address of the `number` variable

Points: 1 / 1

A 37. What will the following code output?

```
int number = 22;
int *var = &number;
cout << var << endl;
```

a. The address of the `number` variable
b. 22
c. An asterisk followed by 22
d. An asterisk followed by the address of the `number` variable

Points: 1 / 1

C 38. What will the following code output?

```
int *numbers = new int[5];

for (int i = 0; i <= 4; i++)
    *(numbers + i) = i;

cout << numbers[2] << endl;
```

a. Five memory addresses
b. 0
c. 3
d. 2
e. 1

Points: 0 / 1

C 39. Look at the following code:

```
int numbers[] = {0, 1, 2, 3, 4 };
int *ptr = numbers;
ptr++;
```

After this code executes, which of the following statements is true?

a. `ptr` will hold the address of `numbers[0]`.
b. `ptr` will hold the address of the 2nd byte within the element `numbers[0]`.
c. `ptr` will hold the address of `numbers[1]`.
d. This code will not compile.

Points: 1 / 1

B 40. To help prevent memory leaks from occurring in C++ 11, a _____ automatically deletes a chunk of dynamically allocated memory when the memory is no longer being used.

a. null pointer
b. smart pointer
c. dereferenced pointer
d. A and C only
e. None of these

Points: 1 / 1