

MULTIPLE CHOICE

1. The _____, also known as the address operator, returns the memory address of a variable.
 - a. asterisk (*)
 - b. ampersand (&)
 - c. percent sign (%)
 - d. exclamation point (!)

2. With pointer variables, you can _____ manipulate data stored in other variables.
 - a. never
 - b. seldom
 - c. indirectly
 - d. All of these

Provide a three-line (or less) C++ statement which emulates your answer for question #2

3. The statement

```
int *ptr;
```

has the same meaning as

- a. `int ptr;`
 - b. `*int ptr;`
 - c. `int ptr*;`
 - d. `int* ptr;`
4. When you work with a dereferenced pointer, you are actually working with:
 - a. a variable whose memory has been deallocated
 - 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

Provide a three-line (or less) C++ statement which emulates your answer for question #4. Start with:

```
int x = 3;
```

5. These can be used as pointers.
 - a. Array names
 - b. Numeric constants
 - c. Punctuation marks
 - d. All of these
 - e. None of these

6. 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

7. A pointer may be initialized with
 - a. the address of an existing object
 - b. the value of an integer variable
 - c. the value of a floating point variable
 - d. all of these

8. 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`.

9. (EXTRA CREDIT) 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

10. (EXTRA CREDIT) 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

11. 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

12. 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

13. Which of the following statements is not valid C++ code (assume `num1` was declared as a float)?

- 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

14. A pointer with the value 0 (zero) is called a NULL pointer.

- a. True
- b. False

15. 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 (`;`)

16. What will the following statement output?

```
Int num1 = 3;  
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

17. A pointer variable is designed to store

- a. any legal C++ value.
- b. only floating-point values.
- c. a memory address.
- d. an integer.
- e. None of these

18. Look at the following statement.

```
int *ptr;
```

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

19. Assuming `ptr` is a pointer variable, what will the following statement output?

```
cout << *ptr;
```

- a. the value stored in the variable whose address is contained in `ptr`.
- b. the string "`*ptr`".
- c. the address of the variable stored in `ptr`.
- d. the address of the variable whose address is stored in `ptr`.

20. 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

21. 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

22. Which statement displays the address of the variable `num1`?

- a. `cout << num1;`
- b. `cout << *num1;`
- c. `cin >> &num1;`
- d. `cout << &num1;`

23. The statement `cin >> *num3;`
- stores the keyboard input into the variable `num3`.
 - stores the keyboard input into the pointer called `num3`.
 - stores the keyboard input into the variable pointed to by `num3`.

Provide an example declaration for the variable `num3` prior to the execution of the statement.

24. Dynamic memory allocation occurs
- when a new variable is created by the compiler
 - when a new variable is created at runtime
 - when a pointer fails to dereference the right variable
 - when a pointer is assigned an incorrect address
25. The statement `int *ptr = new int;`
- results in a compiler error.
 - assigns an integer less than 32767 to the variable named `ptr`.
 - assigns an address to the variable named `ptr`.
 - creates a new pointer named `int`.
26. When using the `new` operator with an older compiler, it is good practice to:
- test the pointer for the NULL address
 - use a preprocessor directive
 - clear the data from the `old` operator
 - All of these
27. Every byte in the computer's memory is assigned a unique
- pointer
 - address
 - dynamic allocation
 - name
28. It is legal to subtract a pointer variable from another pointer variable.
- True
 - False

Justify your answer

29. A pointer variable may be initialized with
- any non-zero integer value within the integer range.
 - any address in the computer's memory allowed by the Operating System.
 - an address less than 0
 - a and c only.
30. If a variable uses more than one byte of memory, for pointer purposes its address is:
- the address of the last byte of storage.
 - the average of the addresses used to store the variable.
 - the address of the first byte of storage.

Explain how this relates to an array of integers

31. What will the following code output?

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

- The address of the `number` variable
- 22
- An asterisk followed by 22
- An asterisk followed by the address of the `number` variable

32. 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

33. 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

34. 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.

35. An array name is a pointer constant because the address stored in it cannot be changed during runtime.

- a. True
- b. False

36. C++ does not perform array bounds checking, making it possible for you to assign a pointer the address of an element out of the boundaries of an array.

- a. True
- b. False

37. A pointer can be used as a function argument, giving the function access to the original argument.

- a. True
- b. False

Explain what this means in terms of scope in terms of the calling function (which could be `main`) and/or the function itself.

38. The ampersand (&) is used to dereference a pointer variable in C++.

- a. True
- b. False

39. 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;
```

- a. True
- b. False