CENTENNIAL COLLEGE

LAB 3

COMP-217 SEC- 001

Name:Muhammad Saad Student Id: 301077320

Instructions:

* This assignment is supposed to be completed individually.
* Add you name and student Id on top of the Lab 3 Doc.
* Please add correct answer under answer section for each question.
* Submit the file with correct responses in the submission box (LAB 3) under Assignments section.
* Each question has 1 mark each.
* Please pick up the answers as a/b/c/d from options (do not add complete text).
* Please add the answers sequentially as per the questions by editing this file.
* For submission name the file with following convention :

**studentfirstname\_lastname\_lab3**

Q1: An array is *not*:

a. A consecutive group of memory locations.

b. Subscripted by integers.

c. Made up of different data types.

d. None of the above.

**ANS C**

Q2: Which of the following is *false*?

a. The last element of an array has position number one less than the array size.

b. The position number contained within square brackets is called a subscript.

c. A subscript cannot be an expression.

d. All of the above.

**ANS A**

Q3: Which statement would be used to declare a 10-element integer array c?

a. array c<12>;

b. array c<int, 12>;

c. array<12> c;

d. array<int, 12> c;

**ANS D**

Q4: Which of the following is *not* a correct way to initialize the array named n?

a. array<int, 5> n{0, 7, 0, 3, 8};

b. array<int, 5> n{0, 7, 0, 3, 8, 2};

c. array<int, 5> n{7};

d. array<int, 5> n{9, 1, 9};

**ANS: B**

Q5: Constant variables:

a. Can be assigned values in executable statements.

b. Do not have to be initialized when they are declared.

c. Can be used to specify array sizes, thereby making programs more scalable.

d. Can be used to specify array sizes, but this makes programs harder to understand.

**ANS C**

**Q6: Referencing elements outside the array bounds with the [] operator:**

a. Can result in changes to the value of an unrelated variable.

b. Is impossible because C++ checks to make sure it does *not* happen.

c. Is a syntax error.

d. Enlarges the size of the array.

**ANS: a**

Q7: Assume that the array named items contains the integer values 0, 2, 4, 6 and 8. Which of the following set of statements uses the range-based for loop to display each value in items?

a. for (int i = 0; i <items.size(); ++i) {  
 cout <<items[i] <<endl;  
}

b. for (int item : items) {  
 cout <<items[item] <<endl;   
}

c. for (int item : items) {  
 cout <<item <<endl;   
}

d. for (int item : items.size()){  
 cout <<item <<endl;   
}

ANS: C

Q8: Which of the following tasks cannot be performed using a range-based for loop?

a. Calculating the product of all the values in an array.

b. Displaying all even element values in an array.

c. Incrementing the value stored in each element of the array.

d. Accessing the element’s subscript.

**ANS: D**

Q9: In order to calculate the \_\_\_\_\_\_\_\_\_\_ of an array of values, the array values must first be summed.

a. Average.

b. Minimum.

c. Maximum.

d. Distribution.

**ANS: A**

Q10: Linear search can be used on:

a. Unsorted arrays.

b. Sorted arrays.

c. Integer arrays.

d. Any of the above.

**ANS: D**

Q11: You can sort an array with the Standard Library’s:

a. sort function.

b. sort\_array function.

c. array\_sort function.

d. None of the above.

**ANS: A**

Q12: A double subscripted array declared as array<array<int, 5>, 3> values; has how many elements?

a. 15

b. 13

c. 10

d. 8

**ANS: A**

Q13: Which of the following does *not* declare a 2-by-2 array and set all four of its elements to 0?

a. array<array<int,2>, 2> b;  
b[0][0] = b[0][1] = b[1][0] = b[1][1] = 0;

b. array<array<int, 2>, 2> b = {0};

c. array<array<int, 2>, 2> b;  
for (auto const &row : b) {  
 for (auto &element : row) {  
 element = 0;  
 }  
}

d. All of the above initialize all four of the array elements to 0.

**ANS: B**

Q14: In a typical *nested* for loop (not a range-based for loop) used to process a two-dimensional array, following the end of each execution of the inner for loop:

a. The outer for loop initializes its counter variable.

b. The outer for loop increments its counter variable.

c. The inner for loop initializes its counter variable.

d. The inner for loop increments its counter variable.

**ANS: B**

Q15: Which of the following is *not* true of class template vector?

a. The size of a vector can be changed after it is declared.

b. A vector can be assigned to another vector by using the assignment operator.

c. A vector object can be initialized with a copy of another vector by invoking the copy constructor.

d. A vector can store only data of type int.

**ANS: D**

Q16: When using exception handling, place any code that might *throw* an exception in a \_\_\_\_\_\_\_\_\_\_.

a. catch block

b. try statement.

c. throw block.

d. what statement.

**ANS: B**

Q17: Which statement about exception handling is *false*?

a. Call the exception object's what member function to get the error message that's stored in the exception object.

b. Bounds checking is performed at execution time with vector member function at, and if a subscript is within the bounds of the array, the member function throws an out\_of\_bounds exception.

c. The catch block contains the code that handles an exception if one occurs.

d. None of the above statements in false.

**ANS: A**

Q18: Pointers *cannot* be used to:

a. Contain memory addresses.

b. Reference values directly.

c. Pass an argument by reference.

d. Manipulate dynamic data structures.

**ANS: A**

Q19: Pointers may be assigned which of the following values?

a. Any integer values.

b. An address.

c. nullptr.

d. Both (b) and (c).

**ANS D**

Q20: What does the following statement declare?

int \*countPtr, count;

a. Two int variables.

b. One pointer to an int and one int variable.

c. Two pointers to ints.

d. The declaration is invalid.

**ANS: B**

Q21 [C++11]: Which of the following statements about pointer initialization and values is *false*?

a. Prior C++11, the value specified for a null pointer was 0 or NULL.

b. When 0 is assigned to a pointer, it’s converted to a pointer of the appropriate type.

c. The value 0 is the *only* integer value that can be assigned directly to a pointer variable without first casting the integer to a pointer type.

d. In the new standard, you should use the constant null\_ptr to initialize a pointer instead of 0 or NULL.

**ANS: C**

Q22: The & operator *can* be applied to:

a. constants.

b. string literals.

c. *lvalues*.

d. *rvalues*.

**ANS C**

Q23: All of the following *can* cause a fatal execution-time error *except*:

a. Dereferencing a pointer that has not been assigned to point to a specific address.

b. Dereferencing a pointer that has not been initialized properly.

c. Dereferencing a null pointer.

d. Dereferencing a variable that is not a pointer.

**ANS D**

Q24: Three of the following expressions have the same value. Which of the following expressions has a value *different* from the others’?

a. \*&ptr

b. &\*ptr

c. \*ptr

d. ptr

**ANS D**

Q25: Which of the following is *not* a valid way to pass arguments to a function in C++?

a. By reference with reference arguments.

b. By value.

c. By reference with pointer arguments.

d. By value with pointer arguments.

**ANS: D**

Q26: When a compiler encounters a function parameter for a single-subscripted array of the form int a[], it converts the parameter to:

a. int a

b. int &a

c. int \*a

d. No conversion is necessary.

**ANS C**

Q27: Which of the following is *not* a correct way to initialize a built-in array?

a. int n[5]{0, 7, 0, 3, 8, 2};

b. int n[]{0, 7, 0, 3, 8, 2};

c. int n[5]{7};

d. int n[5]{9, 1, 9};

**ANS: A**

Q28: Which statement would be used to declare a 10-element integer array c?

a. array c = int[10];

b. c = int[10];

c. int array c[10];

d. int c[10];

**ANS D**

Q29: To *prevent* modification of a built-in array’s values when you pass the built-in array to a function:

a. The built-in array must be declared staticin the function.

b. The built-in array parameter can be preceded by the constqualifier.

c. A copy of the built-in array must be made inside the function.

d. The built-in array must be passed by reference.

**ANS: B**

Q30: Which of the following is *false* about a function to which a built-in array is being passed?

a. It always knows the size of the built-in array that is being passed.

b. It is being passed the address of the first element in the built-in array.

c. It is able to modify the values stored in the built-in array.

d. The built-in array’s name is used as an argument in the function call.

**ANS: C**

Q31[C++11]: Given a *built-in array* of ints named values, which of the following statements would sort the array?

a. sort(values.begin(), values.end());

b. sort(values.array\_begin(), values.array\_end());

c. sort(begin(values), end(values));

d. sort(array\_begin(values), array\_end(values));

**ANS: C**

Q32: A function that modifies an array by using pointer arithmetic such as ++ptr to process every value of the array should have a parameter that is:

a. A nonconstant pointer to nonconstant data.

b. A nonconstant pointer to constant data.

c. A constant pointer to nonconstant data.

d. A constant pointer to constant data.

**ANS: A**

Q33: A function that prints a string by using pointer arithmetic such as ++ptr to output each character should have a parameter that is:

a. A nonconstant pointer to nonconstant data.

b. A nonconstant pointer to constant data.

c. A constant pointer to nonconstant data.

d. A constant pointer to constant data.

**ANS: B**

Q34: Which of the following best describes the array name n in the declaration int n[10];?

a. n is a nonconstant pointer to nonconstant data.

b. n is a nonconstant pointer to constant data.

c. n is a constant pointer to nonconstant data.

d. n is a constant pointer to constant data.

**ANS: C**

Q35: What method should be used to pass an array to a function that does not modify the array and only looks at it using array subscript notation:

a. A nonconstant pointer to nonconstant data.

b. A nonconstant pointer to constant data.

c. A constant pointer to nonconstant data.

d. A constant pointer to constant data.

**ANS: B**

Q36: sizeof:

a. Is a binary operator.

b. Returns the total number of elements in an array.

c. Usually returns a double.

d. Returns the total number of bytes in a variable.

**ANS: D**

Q37: Which of the following gives the number of elements in the array int r[10]?

a. sizeof r

b. sizeof (\*r)

c. sizeof r / sizeof (int)

d. sizeof (\*r) / sizeof (int)

**ANS: C**

Q38: Which of the following *can* have a pointer as an operand?

a. ++

b. \*=

c. %

d . /

**ANS: A**

Q39: Given that k is an integer array starting at location 2000, kPtr is a pointer to k and each integer is stored in 4 bytes of memory, what location does kPtr + 3 point to?

a. 2003

b. 2006

c. 2012

d. 2024

**ANS: C**

Q40: A pointer *can not* be assigned to:

a. Another pointer of the same type without using the cast operator.

b. A pointer to void without using the cast operator.

c. A pointer of a type other than its own type and void *without* using the cast operator.

d. Any other pointer by using the cast operator.

**ANS: D**

Q41: Comparing pointers and performing pointer arithmetic on them is meaningless unless:

a. They point to elements of the same array.

b. You are trying to compare and perform pointer arithmetic on the values to which they point.

c. They point to arrays of equal size.

d. They point to arrays of the same type.

**ANS: A**

Q42: Assuming that t is an array and tPtr is a pointer to that array, which expression refers to the address of element 3 of the array?

a. \*(tPtr + 3)

b. tPtr[3]

c. &t[3]

d. \*(t + 3)

**ANS: C**

Q43: Consider the following function:

void reverse(char \*string1, const char \*string2)

{

int stringsize{sizeof(string1)/sizeof(char)};

\*(string1 + stringsize – 1) = '\0';

string1 = string1 + stringsize – 2;

for (; \*string2 != '\0'; string1--, string2++) {

\*string1 = \*string2;  
 }

}

What technique does the function use to refer to array elements?

a. Array subscript notation.

b. Pointer/offset notation where the pointer is actually the name of the array.

c. Pointer subscript notation.

d. Pointer/offset notation.

**ANS: B**

Q44: Which of the following is *false* for pointer-based strings?

a. A string may include letters, digits and various special characters (i.e., +, -, \*).

b. A string in C++ is an array of characters ending in the null character ('\0').

c. String literals are written inside of single quotes.

d. A string may be assigned in a declaration to either a character array or a variable of type char \*.

**ANS: C**

Q45: cin.getline(superstring, 30);

is equivalent to which of the following?

a. cin.getline(superstring, 30, '\0');

b. cin.getline(superstring, 30, '\n');

c. cin.getline(superstring, 30, '\s');

d. cin.getline(superstring, 30, '\t');

**ANS: B**

Q46: A string array:

a. Stores an actual string in each of its elements.

b. Can only provide access to strings of a certain length.

c. Is actually an array of pointers.

d. Is always less memory efficient than an equivalent double-subscripted array.

**ANS: C**

Q47: Which of the following is *not* true of pointers to functions?

a. They contain the starting address of the function code.

b. They are dereferenced in order to call the function.

c. They can be stored in arrays.

d. They can not be assigned to other function pointers.

**ANS: D**

Q48: (\*max)(num1, num2, num3);:

a. Is the header for function max.

b. Is a call to the function pointed to by max.

c. Is the prototype for function max.

d. Is a declaration of a pointer to a function called max.

**ANS: B**

Q49 function at to retrieve vector elements \_\_\_\_\_\_\_\_\_\_ perform bounds checking.

a. Does not, does not.

b. Does not, does.

c. Does, does not.

d. Does, does.

**ANS: B**