Week 03 Sample Exam

CSE 232 (Introduction to Programming II)

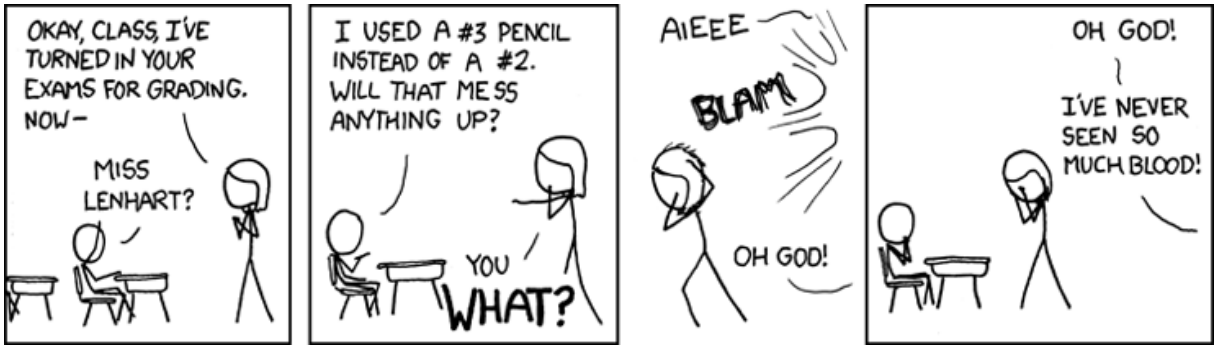
VERSION A

Full Name: .....

Student Number: .....

Instructions:

- DO NOT START/OPEN THE EXAM UNTIL TOLD TO DO SO.
- You may however write and bubble in your name, student number and exam **VERSION/FORM LETTER** (with a #2 pencil) on the front of the printed exam and bubble sheet prior to the exam start. This exam is Version A. Your section doesn't matter and can be ignored.
- Present your MSU ID (or other photo ID) when returning your bubble sheet and printed exam.
- Only choose one option for each question. Please mark the chosen option in both this printed exam and the bubble sheet.
- Assume any needed `#include`s and `using std::...;` namespace declarations are performed for the code samples.
- Every question is worth the same amount of points. There are 55 questions, but you only need 50 questions correct for a perfect score.
- No electronics are allowed to be used or worn during the exam. This means smart-watches, phones and headphones need to be placed away in your bag.
- The exam is open note, meaning that any paper material (notes, slides, prior exams, assignments, books, etc.) are all allowed. Please place all such material on your desk prior to the start of the exam, (so you won't need to rummage in your bag during the exam). Please be sure to bring the required textbook!
- If you have any questions during the exam or when you finish the exam, please raise your hand and a proctor will attend you.



http://xkcd.com/499/ Date Accessed: October 16, 2024

1. Why is it recommended to use `static_cast` to int on the result of the `size()` member function of library containers?

   (a) Because the function returns an unsigned int
   (b) Because it is necessary for compatibility with the C language
   (c) Because for loops required that ints (not bools) be compared in the conditional clause
   (d) Because `size` can return negative values
   (e) Because earlier versions of C++ would raise errors if it wasn't done
   (f) Because `size` is undefined on default constructed containers
   (g) None of the above are true

2. Why should you use the `at` member function on arrays? Example:
```
int array[3] = {1, 2, 3};
cout << array.at(1);
```

   (a) Because it checks the types of its arguments
   (b) Because it can be used on arrays of any size
   (c) Because it returns a reference to the element, not a copy
   (d) Because it is faster than `operator[]`
   (e) Because it returns a null value if the argument is negative
   (f) Because it performs bounds checking
   (g) You shouldn't as it wouldn't compile

3. What is the best way to represent the value 5.7 in the variable `a`?

   (a) `int a{5.7};`
   (b) `int a = 57/10;`
   (c) `int * a = 5.7;`
   (d) `char a = 5.7;`
   (e) `double a = 5.7;`

4. Read the following error message:
```
/tmp/V18paXdMYO.cpp:14:27: error:
invalid conversion from 'const char*'
to 'int' [-fpermissive]
14 | cout << abc("hi") << '\n';
```

   What variable type can be passed into `abc`?

   (a) `char *`
   (b) `std::string`
   (c) Impossible to determine
   (d) `int`
   (e) String literals
   (f) `double`

5. When is it appropriate to use `operator[]` instead of `at` member function for indexing a data structure?

   (a) To reduce the risk of segmentation faults
   (b) To reduce the number of characters used in code (typing `[]` is less shorter than `.at()`)
   (c) When the data structure is a traditional array
   (d) When you need to use negative indicies to index from the end of a structure
   (e) To reduce the number of runtime errors

6. What is the purpose of the `std::string::npos` identifier?

   (a) It is an exception that is thrown when invalid data is given to a string.
   (b) It indicates that an index isnt found in the string.
   (c) It represents an empty string cant be used with that function/method.
   (d) It is the value that is given to a string method when you want to indicate that negative positions/indices should be used.
   (e) It is the index one-past-the-end of a C-style string.

7. How many times will `func` be invoked?

```
vector<int> v = {1, 2, 3};
for (
    auto i = v.size() - 1;
    i >= 0;
    --i) {
  func(v);
}
```

   (a) Impossible to say as the code can't compile.
   (b) More than 3
   (c) 3
   (d) 2
   (e) 1
   (f) 0

8. What does the following program output?

```
vector<int> vec {1, 2, 3};
vec[vec.size()] = 0;
cout << vec.size();
```

   (a) `4`
   (b) Unknown, because it has undefined behavior.
   (c) `3`
   (d) Unknown, because the size of the vector is unsigned.
   (e) None of the above.

9. Why should you use a vector instead of an array to hold a sequence?

   (a) Because using vector is more readable and clear.
   (b) Because using vector is safer.
   (c) Because a vector has many helpful member functions.
   (d) Because a vector can grow to any size at run-time.
   (e) All of the above.

10. Which of the following will copy a directory at path "a" to path "b"?

   (a) `cp -r b a`
   (b) `cp b a`
   (c) `copy -d a b`
   (d) `copy b a`
   (e) `cp -r a b`
   (f) `copy -r a b`
   (g) `copy a b`
   (h) None of the above are correct.

11. What is the type of `x`?

```
string y("hello");
auto x = y.size();
```

   (a) `string::npos`
   (b) `string::size_type`
   (c) `unsigned int`
   (d) `unsigned long`

12. What is the value of `std::string::npos`?

   (a) 0
   (b) The largest value possible in a `string::size_type`.
   (c) 2^{16}
   (d) The size of the string (one past the end).
   (e) None of the above.

13. If you know an integer variable can never become negative (like the size of a small vector), which type do we recommend using to hold such a value?

   (a) `int`
   (b) `unsigned int`
   (c) `char`
   (d) `long`
   (e) `long long`
   (f) None of the above

14. What flag is needed by the `rm` command to delete directories?

   (a) `-c`
   (b) `-r`
   (c) `-f`
   (d) `-d`
   (e) No flag is needed.

15. Why do we recommend using the `at` method instead of `[]` to index into a vector?

    (a) Because the `at` method is faster to type.
    (b) Because the `at` method can be used with negative indexes.
    (c) Because the `at` method is faster.
    (d) Because the `at` method checks for out-of-bounds.
    (e) Because the `at` method returns a reference.
    (f) Because the `at` method returns a value even if the vector is empty.
    (g) None of the above.

16. Some `string` methods return std::string::npos, what does this value mean?

    (a) It means that the method was unable to return an index in the string
    (b) It means that the method threw an exception due to invalid arguments
    (c) It means that the string is uninitialized and hence can't be used
    (d) It means that the string must be empty

17. What advantage do you gain with using the `at` method instead of the access operator (`[]`)?

    (a) It raises an exception instead of invoking undefined behavior if the index is out of bounds
    (b) It is faster as the `at` method is able to directly return the value from the vector's underlying data (the array)
    (c) It works on all containers, whereas the access operator only works on vectors
    (d) It works on all vectors, even if they are const or references
    (e) None of the above

18. Which of the following are good reasons to use a vector instead of an array?

    (a) Vectors can grow at runtime to be any needed size.
    (b) Vectors provide safer methods for access.
    (c) Vectors provide useful methods for manipulation.
    (d) Vectors have similar performance to arrays.
    (e) All of the above.
    (f) All of the except for one of the options.

19. What does the `-r` flag on a `cp` command mean?

    (a) It means to remove the original files after the move.
    (b) It means to return the files to their original location.
    (c) It means to copy all files (including sub-directories) when copying a directory.
    (d) It means to restore the files after deletion.
    (e) It means to copy all the files of a directory in read-only mode.
    (f) None of the above.

20. Why is it common to read from `cin` inside of a while statements conditional (i.e. `while (cin >> x) ...`)?

    (a) So that `cin` is left in a reset state after the loop terminates.
    (b) So that the while loop ends when `x` is no longer positive.
    (c) So that when there is no more input, the while loop automatically ends itself.
    (d) So that the code can easily also ready from other types of `istream`s.
    (e) So that `cin` is never put into an error state from bad input.
    (f) You can't use `cin` in the way described above, doing so will generate a compiler error.

21. Which of the following types is not allowed?

    (a) `vector<int *>`
    (b) `vector<vector<string>>`
    (c) `vector<int **>`
    (d) `vector<int>`
    (e) `vector<int> const`
    (f) All of the above are allowed.

22. Which of the following is **NOT** a fundamental type?

    (a) `int`
    (b) `short`
    (c) `long`
    (d) `unsigned`
    (e) `char`
    (f) `string`
    (g) `float`
    (h) `double`
    (i) All of the above are fundamental types.
    (j) Multiple of the above are **NOT** fundamental types.

23. If the input is "`12 13 4.5 5 cat`" (the double quotes are not part of the input, the 1 is the first character), how many times will the body of the while loop run?

```
int x;
while (cin >> x) {
  ...
}
```

    (a) 0 (b) 1 (c) 2 (d) 3 (e) 4 (f) 5 (g) 6 (h) 7 (i) 8

24. How many times will this for loop's body iterate?

```
string str = "abcd";
for (
  string::size_type i = str.size();
  i >= 0;
  --i)
    // Loop body
```

    (a) 0 (b) 1 (c) 2 (d) 3 (e) 4 (f) 5 (g) > 5

25. When should you use the `at` method (`x.at(1)`) instead of subscripting (`x[1]`) on a string?

    (a) When you need the fastest performance.
    (b) When you know your index is in bounds.
    (c) When you need a check that your call is legal.
    (d) None of the above.
    (e) (a) and (b)
    (f) (b) and (c)

26. What is the value of `x` if the user types the following characters: space character, c, a, t, space character, d, o, g, newline character.

```
string x; cin >> x;
```

    (a) `" cat dog\n"`
    (b) Undefined Behaviour.
    (c) `"cat"`
    (d) `" cat"`
    (e) `" cat dog"`
    (f) `" cat "`
    (g) `""`
    (h) None of the above.

27. What is the type and value of `x`?

```
string str = "abcd";
auto x = str.find('b', 2);
```

    (a) `string::size_type`, `1`
    (b) `string::size_type`, `string::npos`
    (c) `unsigned int`, `1`
    (d) `unsigned int`, `string::npos`
    (e) `int`, `string::npos`
    (f) The code will not compile.
    (g) `int`, `1`
    (h) The code will not run.
    (i) None of the above.

28. `void` functions don't require a return statement. What other functions aren't required to provide a return statement?

    (a) `int main()`
    (b) Functions with default arguments
    (c) Templated functions
    (d) Overloaded functions

29. If you use the Address-Of operator on a pointer to a string, what type is returned?

    (a) A pointer to a pointer to a string
    (b) A const string
    (c) A pointer to a string
    (d) A string
    (e) A pointer to a const string
    (f) None of the above

30. If you declare a string (e.g. `string x;`), what is its initial value?

    (a) It depends on if the string is dynamically allocated
    (b) Undefined (or compiler dependant)
    (c) The empty string
    (d) A random value

31. Where are files removed by the `rm` command temporarily stored for 30 days?

    (a) `/.deleted`
    (b) `~/.trash`
    (c) `/mnt/c/RecycleBin`
    (d) `/trash`
    (e) None of the above

32. What will the following code print?

```
std::string word = "TEST";
for (auto & letter : word) {
    letter = letter - 'A' + 'a';
}
std::cout << word << std::endl;
```

    (a) `test`
    (b) `t`
    (c) `uftu`
    (d) `TEST`
    (e) Nothing; you cannot perform math on characters.

33. If I have an `std::string` called `name`, with the contents "John Wilkes Booth". Which of the following techniques will set `middle` to be "Wilkes"?

    (a) `middle = name[6]+name[7]+name[8]+name[9]+name[10]+name[11];`
    (b) `middle = name.substr(5,6);`
    (c) `middle = name.substr(6,6);`
    (d) `middle = name[5]+name[6]+name[7]+name[8]+name[9]+name[10];`
    (e) `middle = name.find(' ', ' '));`
    (f) More than one of the above.

34. What would be the value of `pos2` after the following C++ code?

```
std::string str{"One,Two,Three"};
size_t pos1 = str.find(',');
size_t pos2 = str.find(',', pos1);
```

    (a) `0`
    (b) `1`
    (c) `3`
    (d) `7`
    (e) `13`
    (f) `std::npos`

35. What will the following code output?

```
std::string letter_set="while";
for (char & let : letter_set) {
    let -= 2;
}
std::cout << letter_set << std::endl;
```

    (a) `while`
    (b) `wwwww`
    (c) `ufgjc`
    (d) `aaaaa`
    (e) None of the above

36. A common idiom for reading in values from standard input is used below. When will this loop terminate?

```
std::string s;
while (cin >> s) {
        ...
}
```

(a) Never, the loop will run forever
(b) When the End-Of-File is encountered
(c) When a integer or floating point value is encountered
(d) When punctuation is encountered
(e) When whitespace is encountered

37. What is the value of `x` after this code runs?

```
string x = "abcd";
x.at(1) = "XYZ";
```

(a) "aXYZbcd"
(b) "abcd"
(c) "XYZabcd"
(d) "aXYZcd"
(e) "XYZbcd"
(f) The code won't compile.

38. What is the output from the following code?

```
std::string str1 = "statistic";
std::string str2 = "provision";
for (int i = 0; i < str1.size(); ++i)
{
    if (str1[i] == 't') continue;
    std::cout << str2[i];
}
```

(a) `poison`
(b) `prvsin`
(c) `vision`
(d) `provis`

39. Files deleted with the `rm` command can be restored from accidental deletion how?

(a) There is no way to recover the lost files.
(b) By using the `undo` command.
(c) From the backup made in the `.trash` directory.
(d) By running `rm --undo`.

40. The `cp` command copies files. What command is used to copy directories?

(a) `cp -r`
(b) `cpdir`
(c) `cp_dir`
(d) `mv`

41. What is the value of `x` in the following program?

```
auto x = 'b' - 'a';
```

(a) It is the ASCII value for the character `'c'`
(b) `'a'`
(c) `1`
(d) `-1`
(e) `'b'`

42. What is the primary reason we recommend using the `.at` method instead of `[]` for indexing into containers like std::string?

(a) To make out of range indexing an error instead of undefined behavior.
(b) Because the `.at` method can accept signed `int`s, but `[]` only takes `unsigned int`s.
(c) We don't recommend the `.at` method more as the two are equivalent.
(d) The `.at` method can be used on empty strings.

43. If you declare a string, what is its initial value?

(a) 0
(b) false
(c) Empty
(d) Undefined
(e) None of the above.

44. What flag is required to for the mv command to move folders?

(a) None of the above
(b) -r
(c) -d
(d) -f

45. What is the type of x in the code below?

```
// ...
const std::string & xs = thing;
for (auto x : xs) {
        // ...
}
// ...
```

(a) `char`
(b) `std::string`
(c) `int`
(d) Impossible to determine with the information provided

46. What is the output of the following code?

```
std::cout << static_cast<int>('b');
```

(a) 0
(b) 1
(c) 2
(d) 3
(e) 4
(f) None of the above

47. Which of the following is a `char` literal?

(a) `"Z"`
(b) `char_literal`
(c) `'#'`
(d) `char`
(e) `*chr`
(f) `a`

48. If a function's sole purpose is its side-effects, what should its return type be?

(a) It doesn't matter as the return statement won't have a value.
(b) `void`
(c) Omitted
(d) `int`
(e) It depends on if the function performs IO operations.
(f) Impossible to determine as all functions must return a value.

49. What happens if you index beyond the end of a vector. Example:

```
std::vector<int> v{1, 2, 3};
std::cout << v[3];
```

(a) A int will be returned
(b) Undefined Behavior
(c) A Run-Time Error
(d) A Syntax Error
(e) The last int will be returned
(f) An int from another vector will be returned

50. When will a stream (like `cin`) evaluate as false in a conditional expression?

(a) When the stream is in an error state like at End-Of-File.
(b) When the stream is waiting to access a file
(c) Never, a stream is always considered true
(d) Always, a stream is always considered false
(e) When the stream has extracted a false value
(f) When the stream has more characters to process

51. The `-r` flag for the `rm` program is short for what?

(a) relevant
(b) return
(c) recent
(d) repeat
(e) recursive
(f) reverse
(g) ranked
(h) rotating

52. What is printed by the following code?
```
char c = 'e';
c++;
std::cout << c;
```

(a) `f`
(b) `c`
(c) `1`
(d) `e1`
(e) `e`
(f) `d`
(g) None of the above due to a compilation error

53. What does the `std::endl` object do when passed as the second operand to the insertion operator (`<<`)?

(a) It causes the stream to separate words according to whitespace
(b) It resets the stream
(c) It indicates the End-Of-File
(d) It indicates that the stream should be prepared for new characters
(e) It doesn't do anything, it is instead used to indicate comments
(f) It can't be used with an insertion operator
(g) It causes a newline character to be written to the stream

54. What is the type of `x`?
```
std::string const a{"CSE232"};
auto x = a.at(1);
```

(a) `std::string &`
(b) `std::string const &`
(c) `char const`
(d) `std::string const`
(e) `char const *`
(f) `char`
(g) `std::string const *`
(h) `char &`
(i) `char const &`

55. How do you make a variable that can have multiple types?

(a) By using `typedef`
(b) By using `static_cast`
(c) By using `decltype`
(d) By declaring its type as `auto`
(e) It is impossible

56. What is the return type of this function?
```
std::string Exclamation(int num) {
    return "!";
}
```

(a) `int *`
(b) `std::string`
(c) `Exclamation`
(d) `int`
(e) `"!"`
(f) None of the above

57. What is the type of x?
```
auto x = My_Function("abcd");
```

(a) `int`
(b) `void`
(c) `char`
(d) `std::string`
(e) Impossible to determine with the information given

58. The 2 argument version `getline` function will read from a stream until what type of character is encountered?

(a) The newline character
(b) The space character
(c) A whitespace character (include space, newline, and tab)
(d) A non-ASCII character
(e) A non-alphabetic character

59. What is the value of x?
```
std::string s{"abcde"};
int x = static_cast<int>(s.size());
```

(a) 5
(b) 6
(c) 7
(d) int
(e) unsigned int
(f) Impossible to determine with the information given

60. The put to operator (<<) supports chaining. For example:
`cout << "the" << x << 'c' << endl;`
Which other operator(s) support chaining?

(a) operator=
(b) operator>>
(c) operator++
(d) (a) and (b) support chaining
(e) (b) and (c) support chaining
(f) (a) and (c) support chaining
(g) (a), (b), and (c) support chaining

61. Which of the following std::strings is larger (using operator>) than x, if x's value is std::string{"cse"}?

(a) "CSE"
(b) "bus"
(c) "cs"
(d) "ee"
(e) None of the above

62. How many times will isspace be invoked?
```
vector<int> v = {1, 2, 3};
for (
    auto i = 0;
    i < v.size();
    ++i) {
  isspace(' ');
}
```

(a) 0
(b) 1
(c) 2
(d) 3
(e) More than 3
(f) Impossible to say as the code can't compile.

63. Which of the following expressions will change a lowercase letter stored in the variable named x to be upper case?

(a) x = x - 'a' + 'A'
(b) x = x - 'A' + 'a'
(c) x = toupper(x)
(d) x *= 2
(e) Two of the above will perform the conversion.
(f) Three of the above will perform the conversion.
(g) Four of the above will perform the conversion.
(h) None of the above will perform the conversion.

64. Which of the following is NOT a benefit to using vectors instead of arrays?

(a) Vectors are part of the C library
(b) Vectors can change and report their size
(c) Vectors have useful member functions
(d) Vectors can change their size at runtime
(e) All of the above are true

65. What will the following code do?
```
vector<int> vec{2, 3};
cout << array.at(2);
```

(a) It will raise a compile-time error
(b) It will output a `3`
(c) It will raise a run-time error
(d) It will output a `2`
(e) It will do two of the above options

66. Below I have the contents of a file named "main.cpp". But it won't compile, what needs to be changed to allow this program to compile?
```
#include<iostream>
int main() {
  my_print(4.5);
}
int my_print(int a) {
  std::cout << a;
  return 3;
}
```

(a) The main function needs a return statement.
(b) The function needs to be declared before the main function.
(c) The argument (`4.5`) needs to be changed to an integer, like (`4`).
(d) The function's return type should be changed to `void`.
(e) The name of the function needs to be changed to camel-case, `MyPrint`.
(f) Two of the above options needs to be implemented for the program to compile.

67. The following code generates a compiler error, what is the cause of the error?
```
string title{"Tour of C++"};
string other;
for (char const & letter : title) {
  other.push_back(letter);
}
```

(a) The for loop is written incorrectly.
(b) `other` in uninitialized.
(c) `title` doesn't have the required null character.
(d) The `push_back` member function can't be called on an empty string.
(e) `title` has invalid characters within it.
(f) The variable `letter` is an invalid type.
(g) The code above should compile without any changes needed.

68. What value is returned by string's find member function if the specified character isn't found in the string?

(a) `std::string::size_type`
(b) `0`
(c) `nullptr`
(d) `size_t`
(e) `std::string::npos`
(f) None of the above

69. Which of the following types can be concatenated with a std::string?

(a) `std::string`
(b) C-style strings
(c) `char`
(d) String literals
(e) All of the above

70. By default, the gets from operator (`>>`) ignores which characters?

(a) Digits
(b) Punctuation
(c) End-Of-File
(d) Whitespace
(e) All of the above

71. Why should care be taken when using the `rm` Unix command?

    (a) Because it will duplicate files found in the home directory.
    (b) Because it can cause segmentation faults.
    (c) Because files deleted by it can't be restored.
    (d) Because the `rm` can only be used on C++ source files.
    (e) None of the above

72. Which of the following statements are true about C-style strings?

    (a) `"word"` is an example of a C-style-string literal
    (b) It is an array of char
    (c) They are different from `std::string`
    (d) The last character is always the null character
    (e) All of the above are true

73. The -r option when provided to the `rm` causes what change in behavior?

    (a) It now can rename files
    (b) It now will give informative suggestions
    (c) It now can prevent unwanted edits
    (d) It now can connect to the internet
    (e) It now can reverse changes to the file system
    (f) It now can delete directories
    (g) It now can change the contents of files

74. What is the value of `title` after the following code?

```
string title{"CSE232"};
for (char letter : title) {
  letter = 'x';
}
```

    (a) `"x"`
    (b) `"CSE232"`
    (c) `""`
    (d) `"letter"`
    (e) `"'x''x''x''x''x''x'"`
    (f) `"xxxxxx"`
    (g) None of the above because the above code won't compile

75. What's the best way to print a double 'a' with exactly 2 places after the decimal point?

    (a) `cout << std::to_string(a, 2);`
    (b)
```
std::string x =
std::to_string(a);
cout << a.substr(0, a.find('2')+
3);
```
    (c) `cout << fixed << setprecision(2) << a;`
    (d) `printf("%2f", a);`
    (e) `cout << a.places(2);`

76. Why doesn't this code make the string uppercase?

```
std::string s{"Hello"};
for (char c : s)
  c = toupper(s);
```

    (a) `toupper` isn't a function in the standard library
    (b) The `for` statement doesn't have a body
    (c) The code does actually make the string uppercase
    (d) It is impossible to change the string as strings are immutable
    (e) `string s` isn't initialized
    (f) `c` is a copy of each element

77. C-style strings always end with which character?

(a) The null character
(b) The character whose value is false
(c) `'\0'`
(d) The character that has the ASCII value of 0
(e) All of the above are true

78. What would the following code output?
```
vector<int> vec {10, 20, 30};
cout << vec.at(3);
```

(a) It would output the value of memory one-past-the-end of the vector
(b) `30`
(c) `40`
(d) A runtime error would be raised
(e) Impossible to determine given undefined behavior
(f) `3`
(g) None of the above are true

79. What is the value of `b` in the following expression?
```
bool b{string::npos};
```

(a) A very large number
(b) The above code wouldn't compile
(c) `false`
(d) `true`
(e) Impossible to determine given implementation-specific behavior

80. Which of the following statements will increase the size of an array of ints named `arr`?

(a) `vector<int> arr2{arr};`
(b) `arr = arr + 1;`
(c) `arr += 1;`
(d) `arr++;`
(e) `arr.resize(arr.size() + 1);`
(f) None of the above

81. What will the following terminal command do (assuming dir_a and dir_b are directories/folders)?
```
cp dir_a dir_b
```

(a) dir_a will be copied into dir_b
(b) dir_b will be copied into dir_a
(c) Nothing, it will raise an error
(d) The contents of dir_b will be copied into dir_a
(e) The contents of dir_a will be copied into dir_b

82. The body of this if statement will run under which conditions:
`if (cin >> x) ...`)?

(a) When the EndOfFile has been reached
(b) When `x` has been assigned a value from standard input
(c) The body will always run because `cin >> x` always has a true value
(d) It is impossible to determine without knowing the contents of the body
(e) The body will never run because `cin >> x` is not a boolean value
(f) When a value that can't be put into `x` has been encountered

83. How many times will this for loop's body iterate?
```
string str = "abcd";
for (
  string::size_type i = 0;
  i <= str.size();
  ++i)
    // Loop body
```

(a) 0 (b) 1 (c) 2 (d) 3 (e) 4 (f) 5 (g) > 5

84. Which of the following converts the char value '7' to the int value 7?

(a) `static_cast<int>('7')`
(b) `(int) '7'`
(c) `int('7')`
(d) `std::to_int('7')`
(e) All of the above
(f) None of the above

85. Which of the following member functions work on both arrays and vectors?

    (a) `[]`
    (b) `at`
    (c) `+`
    (d) `size`
    (e) `&&`
    (f) `*`
    (g) None of the above

86. What would be the value of `y` after the following C++ code?

```
string line{"banana"};
size_t x = line.find('n');
size_t y = line.find('n', x + 1);
```

    (a) `0`
    (b) `1`
    (c) `2`
    (d) `3`
    (e) `4`
    (f) `5`
    (g) `-1`

87. How many parameters does this function have?

```
std::string Fun(int num, string str) {
    return "!";
}
```

    (a) `0`
    (b) `1`
    (c) `2`
    (d) `3`
    (e) `4`
    (f) `5`
    (g) `> 5`

88. Which of the following values will compare equal to `string("Mal")`?

    (a) `string("Mal")`
    (b) `string("mal")`
    (c) `string("MAL")`
    (d) `string("Malcolm")`
    (e) `string("Mal*")`
    (f) All of the above

89. For the code at the end of Section 11.4, if the standard input was the following, what would be the size of `v`?

```
1 2 3 skip 4 5 6 7 stop 8
```

    (a) `1`
    (b) `2`
    (c) `3`
    (d) `4`
    (e) `5`
    (f) `6`
    (g) `7`
    (h) `8`

90. What is the value of `title` after the following code?

```
string title{"CSE232"};
for (char & letter : title) {
  letter = 'x';
}
```

    (a) `"CSE232"`
    (b) `"xxxxxx"`
    (c) `"'x''x''x''x''x''x'"`
    (d) `"x"`
    (e) `""`
    (f) `"letter"`
    (g) None of the above because the above code won't compile

91. Which of the following strings are guaranteed to have size 0?

    (a) `string str = "abcd";`
    (b) `str.at(0) = 0;`
    (c) `string str;`
    (d) `string str = empty;`
    (e) `str.size() = 0;`
    (f) None of the above

92.
```
/tmp/V18paXdMYO.cpp:14:27: error:
invalid conversion from 'const char*'
to 'int' [-fpermissive]
14 | cout << abc("hi") << '\n';
```

    What does the number 14 mean in the above error partial message?

    (a) This is the 14th error resulting from this compilation
    (b) The error is on line 14 of the C++ code
    (c) It is impossible to determine given the partial error message
    (d) The error is at warning level 14 (which means it is fairly unimportant)
    (e) That there have been 14 lines of output before the error was encountered.

93. Why should you not compare unsigned and signed ints directly?

    (a) Because they have different ranges and thus one will have to be implicitly converted to the other
    (b) Because unsigned ints don't support comparison operators until they are `static_cast`ed to int values
    (c) Because unsigned ints are meant to store addresses and sizes, while ints are meant to hold values that can have negative qualities
    (d) Because signed overflow is undefined behavior
    (e) Because doing so will result in a rumtime error that must be resolved before the program can proceed
    (f) Because unsigned ints can't be negative, so comparisons with signed ints is undefined behavior
    (g) None of the above are true

94. Why do you need `#include<string>` but not `#include<int>` to use strings and ints?

    (a) Because there are different versions of string
    (b) Because strings are composed on chars, but ints don't have smaller components
    (c) Because string was not present in the C language
    (d) Because strings are not necessary in every program
    (e) Because string is an aggregate type
    (f) Because string is not a fundamental type
    (g) None of the above

95. What value does the `string::find` method return if the searched for value isn't found?

    (a) `std::string::npos`
    (b) An error
    (c) An empty string
    (d) `false`
    (e) `0`

96. Why do we recommend using the `at` method instead of `[ ]` to index into an array?

    (a) Because the `at` method checks for out-of-bounds.
    (b) Because the `at` method returns a reference.
    (c) Because the `at` method can be used with negative indexes.
    (d) Because the `at` method returns a value even if the array is empty.
    (e) Because the `at` method is faster to type.
    (f) Because the `at` method is faster.
    (g) We don't, because it wouldn't compile

97. What is the output from the following statement?
    ```
    cout << 'c' - 'a';
    ```

    (a) Nothing, the code won't compile
    (b) `b`
    (c) `2`
    (d) `c-a`
    (e) It is the ASCII value for the character `'b'`
    (f) `'b'`