Sample Exam Week 03

CSE 232 (Introduction to Programming II)

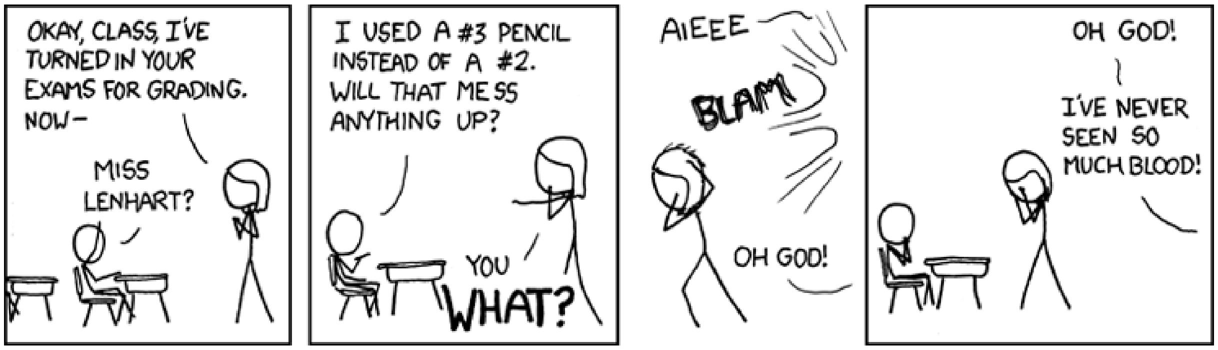
VERSION A

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 NUMBER** (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 `#includes` 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).
- If you have any questions during the exam or finish the exam early, please raise your hand and a proctor will attend you.


OKAY, CLASS, I'VE TURNED IN YOUR EXAMS FOR GRADING. NOW—
MISS LENHART?
I USED A #3 PENCIL INSTEAD OF A #2. WILL THAT MESS ANYTHING UP?
YOU WHAT?
AIEEE
BLAM
OH GOD!
OH GOD!
I'VE NEVER SEEN SO MUCH BLOOD!
http://xkcd.com/499/

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

2. 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

3. What does the following program output?

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

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

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

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

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

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

6. What is the type of `x`?

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

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

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

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

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

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

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

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

11. 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 must be empty
    (d) It means that the string is uninitialized and hence can't be used

12. 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 works on all vectors, even if they are const or references
    (c) It works on all containers, whereas the access operator only works on vectors
    (d) It is faster as the `at` method is able to directly return the value from the vector's underlying data (the array)
    (e) None of the above

13. 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.

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

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

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

16. Which of the following types is not allowed?

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

17. 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.

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

19. 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   (d) 3   (g) > 5
    (b) 1   (e) 4
    (c) 2   (f) 5

20. 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)

21. 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"`
    (b) `""`
    (c) `" cat dog"`
    (d) `" cat "`
    (e) `"cat"`
    (f) Undefined Behaviour.
    (g) `" cat dog\n"`
    (h) None of the above.

22. What is the type and value of `x`?
```
string str = "abcd";
auto x = str.find('b', 2);
```

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

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

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

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

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

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

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

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

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

27. What will the following code print?
```
std::string word = "TEST";
for (auto & letter : word) {
    letter = letter - 'A' + 'a';
}
std::cout << word << std::endl;
```

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

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

29. 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`

30. 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) `aaaaa`
   (c) `ufgjc`
   (d) `wwwww`
   (e) None of the above

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

32. What is the value of `x` after this code runs?
```
string x = "abcd";
x.at(1) = "XYZ";
```
   (a) "abcd"
   (b) "aXYZcd"
   (c) "XYZbcd"
   (d) "XYZabcd"
   (e) "aXYZbcd"
   (f) The code won't compile.

33. 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) `vision`
   (b) `prvsin`
   (c) `poison`
   (d) `provis`

34. 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 running `rm --undo`.
   (c) From the backup made in the `.trash` directory.
   (d) By using the `undo` command.

35. The `cp` command copies files. What command is used to copy directories?
   (a) `cpdir`
   (b) `cp_dir`
   (c) `cp -r`
   (d) `mv`

36. What is the value of `x` in the following program?
`auto x = 'b' - 'a';`

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

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

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

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

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

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

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

40. What is the type of x in the code below?
```
// ...
const std::string & xs = thing;
for (auto x : xs) {
        // ...
}
// ...
```

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

41. 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

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

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

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

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

44. 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) The last int will be returned
(d) An int from another vector will be returned
(e) A Syntax Error
(f) A Run-Time Error

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

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

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

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

47. What is printed by the following code?

```
char c = 'e';
c++;
std::cout << c;
```

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

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

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

49. What is the type of `x`?

```
std::string const a{"CSE232"};
auto x = a.at(1);
```

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

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

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

51. What is the return type of this function?

```
std::string Exclamation(int num) {
    return "!";
}
```

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

52. What is the type of x?

```
auto x = My_Function("abcd");
```

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

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

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

54. 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

55. 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

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

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

57. 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.

58. 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.

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

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

60. 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 output a `2`
(d) It will raise a run-time error
(e) It will do two of the above options

61. 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 name of the function needs to be changed to camel-case, `MyPrint`.
(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 main function needs a return statement.
(f) Two of the above options needs to be implemented for the program to compile.

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

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

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

64. 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

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

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

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

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