# Sample Exam Week 11

## CSE 232 (Introduction to Programming II)

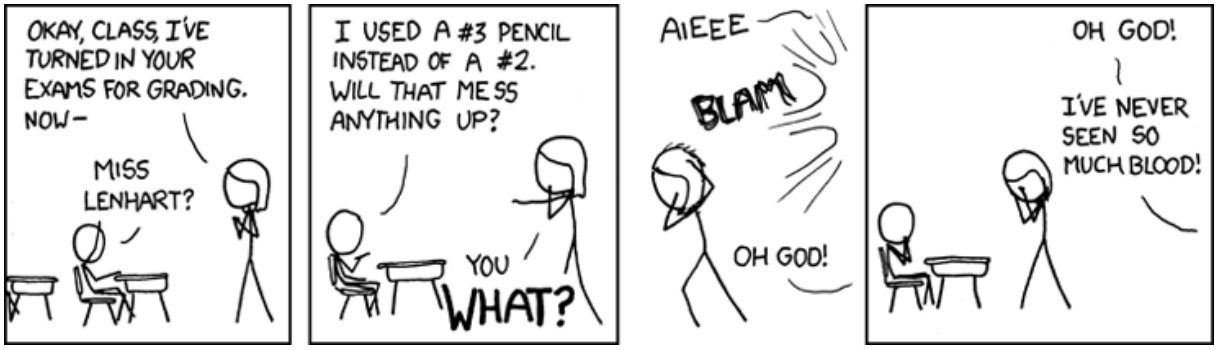
## VERSION A

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




http://xkcd.com/499/

1. Which of the following algorithms can work with data stored in arrays?

   (a) `std::transform`
   (b) `std::sort`
   (c) `std::find_if`
   (d) `std::copy`
   (e) `std::remove`
   (f) `std::accumulate`
   (g) All of the above.

2. Assuming `vec` is a `vector<T>`, what is this code doing?

```
copy(
    vec.begin(),
    vec.end(),
    ostream_iterator<T>(x, ",")
);
```

   (a) Adding a comma to each element of the vector (except for the last)
   (b) Writing the number of commas in a string to a file.
   (c) Converting the vector to a string
   (d) Duplicating a vector
   (e) Writing to a stream named `x`
   (f) None of the above

3. Why should you generally use STL algorithms instead of implementing your own with loops?

   (a) STL algorithms are often more generic.
   (b) STL algorithms are often better tested for correctness.
   (c) STL algorithms are often faster.
   (d) STL algorithms are often more readable.
   (e) All of the above.

4. Which of the following iterators support the decrement operator?

   (a) Bidirectional iterators
   (b) Forward iterators
   (c) Random access iterators
   (d) (a) and (b)
   (e) (b) and (c)
   (f) (a) and (c)
   (g) (a), (b) and (c)
   (h) None of the above

5. Which of the following statements are true, assuming `v1` and `v2` are both non-empty `vector<int>`s?

```
std::copy(v1.begin(), v1.end(),
v2.begin());
std::copy(v1.begin(), v1.end(),
back_inserter(v2));
```

   (a) The first statement requires that `v2.size() >= v1.size()` .
   (b) The second statement enlarges `v2` as needed.
   (c) The first statement will replace existing elements of `v2`.
   (d) The second statement appends each element of `v1` to the end of `v2` in order.
   (e) Two of (a-d) are true.
   (f) Three of (a-d) are true.
   (g) Four of (a-d) are true.

6. Can the `std::sort` function sort an array of `std::string`?

   (a) No, because only `vector`s can be sorted by `std::sort`.
   (b) No, because arrays do not have member functions.
   (c) Yes, because a pointer to the start of the array is all that is needed.
   (d) Yes, as long as the size of the array is known.
   (e) None of the above are valid explanations.

7. I have a string with all sorts of characters in it. I want to convert the string to have all the uppercase letters be converted to lowercase letters. What STL algorithm would be most useful for this conversion?

   (a) `std::transform`
   (b) `std::search`
   (c) `std::accumulate`
   (d) `std::find`
   (e) `std::copy`
   (f) `std::swap`
   (g) `std::count`

8. Which of the following classes / functions are used to allow generic algorithms to output to `std::cout`?

   (a) stream iterators
   (b) string streams
   (c) back inserters
   (d) (a) and (b) both work
   (e) (a) and (c) both work
   (f) (b) and (c) both work
   (g) All of (a), (b), and (c) work
   (h) None of the above

9. Because generic algorithms can't change the size of the containers they work on, how do they indicate which elements are valid (for instance, after a call to `remove_if`?

   (a) They change a non-const argument to an index
   (b) They return an iterator
   (c) They change the invalid elements to a different value
   (d) They call the erase member function

10. What type of iterator is an array's name?

    (a) Random Access Iterator
    (b) Bi-directional Iterator
    (c) Forward Iterator
    (d) All of the above
    (e) None of the above

11. What happens if you use the address-of (&) operator on an iterator?

    (a) Undefined behaviour (it may crash your program).
    (b) It returns the address of the object pointed at.
    (c) It returns the address of the iterator in memory.
    (d) Syntax error, you can't use & on an iterator.
    (e) None of the above.

12. What operations can't be performed on a reverse iterator, but can with a forward iterator?

    (a) `operator++`
    (b) `operator--`
    (c) `operator==`
    (d) `operator*`
    (e) All of the above.
    (f) None of the above.

13. Which of the following algorithms is commonly used to output a data structure to an ostream?

    (a) `std::print_all`
    (b) `std::swap`
    (c) `std::copy`
    (d) `std::generate`
    (e) `std::for_each`
    (f) None of the above.

14. Can `std::sort` be used to sort an array of floats?

    (a) Yes, you just need a pointer to the start and one-past-the-end of the array.
    (b) No, sort requires random access iterators.
    (c) It depends on the compiler.
    (d) No, sort can only be performed on STL containers.

15. Assuming `v` is a `vector<pair<string, int>>` which was filled from a map, what does the following algorithm do?

```
sort(v.begin(), v.end(),
  [](auto a, auto b) {
      return a.second > b.second;
  }
);
```

(a) It sorts the elements by value in descending order.
(b) It sorts the elements by value in ascending order.
(c) It sorts the elements by key in descending order.
(d) It sorts the elements by key in ascending order.
(e) The following code doesn't compile (syntax error).

16. What is the output from the following code:

```
vector<string> points = {
  "10",
  "200",
  "3",
};
vector<int> result;
transform(
  points.begin(),
  points.end(),
  back_inserter(result),
  [](auto const & x) {
    return stoi(x);
  }
);
copy(result.begin(), result.end(),
  ostream_iterator<int>(cout, ","));
```

(a) `10,200,3,`
(b) `10, 210, 213,`
(c) Undefined Behaviour
(d) Compiler Error
(e) `10, 200, 3,`
(f) `213,`
(g) Run-time Error

17. What does the set named `c` contain after the following code?

```
set<int> a = {1, 1, 6, 3, 4};
vector<int> b = {2, 3, 4, 5};
set<int> c = {8, 9};
set_difference(
  a.begin(), a.end(),
  b.begin(), b.end(),
  inserter(c, c.end()));
```

(a) {3, 4, 8, 9}
(b) {8, 9}
(c) {2, 5, 8, 9}
(d) {1, 1, 2, 3, 3, 4, 4, 5, 6, 8, 9}
(e) {1, 1, 6, 8, 9}
(f) {1, 6, 8, 9}
(g) None of the above (it won't compile)

18. `std::find` returns an iterator pointing at the element that was found. What does the algorithm return if a matching element is not found?

(a) `nullptr`
(b) The boolean value `false`
(c) It depends on the type of the container
(d) An iterator to one past the last element in the range

19. Which of the following is an anonymous function often defined at the location that it is invoked or passed as an argument to a function?

(a) A ternary expression
(b) A forward iterator
(c) A generic function
(d) A lambda expression
(e) A predicate function
(f) A member function

20. What property must the containers being used as input for a set algorithm (like `std::set_difference`) satisfy?

    (a) They must be sorted
    (b) They must be sets
    (c) They must be non-empty
    (d) They must not have duplicates
    (e) None of the above