Sample Exam Week 08
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.

http://xkcd.com/499/

---
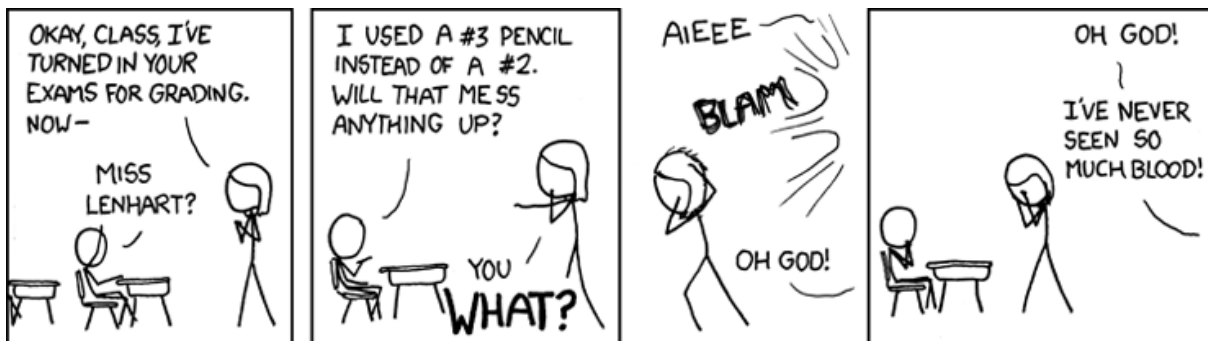
1. What is the name of the operation update a branch with the changes made to the local repository?

   (a) save
   (b) branch
   (c) commit
   (d) push
   (e) squash

2. Which git command is needed to send your local repo's changes/commits to a remote repo (like one hosted on GitHub)?

   (a) `git pull`
   (b) `git add`
   (c) `git push`
   (d) `git commit`
   (e) None of the above.

3. What is the primary situation (at least in this course) when the Rule of Three functions should be implemented?

   (a) When the class has dynamically allocated memory.
   (b) When the class has a default constructor.
   (c) When the class has private data members.
   (d) (a) and (b)
   (e) (a) and (c)
   (f) (b) and (c)
   (g) All of the above.
   (h) None of the above.

4. Which of the following options allow you to write output from your program to a file?

   (a) Command line IO redirection
   (b) `git add`
   (c) `std::ofstream`
   (d) Wildcards
   (e) All of the above
   (f) (a) and (b)
   (g) (a) and (c)
   (h) (b) and (c)
   (i) (a) and (b) and (c)

5. Why should you often mark converting constructors as `explicit`?

   (a) To avoid unintended implicit casts from your class to other types.
   (b) Because converting constructors must be marked explicit by the language standard.
   (c) To avoid unintended implicit casts to your class.
   (d) All of the above.

6. What is returned by the following expression?

   ```
   cout << 'x'
   ```

   (a) The character `'x'`
   (b) `cout`
   (c) `<<`
   (d) `istream &`
   (e) The variable `x`
   (f) None of the above

7. Why should you mark most single argument constructors *explicit*?

   (a) To ensure that programmers know to use it carefully
   (b) It isn't needed, the compiler will do it for you
   (c) You should always mark it explicit, no exceptions
   (d) To stop implicit type conversions
   (e) It stops the compiler from using the default, synthetic function

8. Which of the following git commands involve communicating with a remote repository?

   (a) `git push`
   (b) `git fetch`
   (c) `git clone`
   (d) All of the above

9. In the following (partially included) class, what does the `delete` mean?

```
class Thing {
 public:
  Thing(Thing const &) = delete;
  ...
};
```

   (a) That the objects of this class can't be copied
   (b) That objects of this class have a default constructor
   (c) That objects of this class can't be stored in an array
   (d) That objects of this class will be deleted when they fall out of scope
   (e) None of the above

10. Which of the following is true about the "Rule of Three" member functions?

   (a) When one of the 3 has a custom implementation, then the others **must** have custom implementations.
   (b) When one of the 3 has a custom implementation, then the others **may** also need custom implementations.
   (c) When one of the 3 has a custom implementation, then the others **must not** have custom implementations.
   (d) None of the above

11. Which of the following is a situation where having a custom destructor is likely warranted?

   (a) When the object owns dynamically allocated memory
   (b) When the object has a custom default constructor
   (c) When the object has a custom assignment operator
   (d) (a) and (b)
   (e) (a) and (c)
   (f) (b) and (c)
   (g) (a), (b), and (c)
   (h) None of the above

12. Which of the following is true about streams?

   (a) All streams support both the insertion and extraction operators
   (b) Streams can only be used by a single function
   (c) Streams can't be copied
   (d) Streams can only be used to hold strings

13. Why should the `operator<<` take a reference to `ostream` instead of `ostringstream`?

   (a) So that the function can be chained
   (b) So that is can work will all types of ostreams
   (c) Because you can't make a reference to an ostringstream
   (d) Because using ostringstream won't compile

14. Which of the following operators does `istringstream` support?

   (a) `operator<<`
   (b) `operator>>`
   (c) `operator++`
   (d) (a) and (b)
   (e) (a) and (c)
   (f) (b) and (c)
   (g) (a), (b), and (c)
   (h) None of the above

15. Which of the following tasks benefit from using the `sstream` library?

   (a) Converting a string to a long.
   (b) Converting a long to a string.
   (c) Concatenating multiple strings together.
   (d) Parsing text into its components.
   (e) Formatting lines prior to output.
   (f) All of the above.
   (g) All of the above except (b).

16. Which of the functions must be invoked by the following code?
```
Type x;
x = y;
```
    (a) `Type`'s default constructor
    (b) `Type`'s assignment operator
    (c) `Type`'s copy constructor
    (d) `Type`'s destructor
    (e) `Type`'s conversion constructor
    (f) None of (a-e).
    (g) 2 of (a-e).
    (h) 3 of (a-e).
    (i) 4 of (a-e).
    (j) All of (a-e).

17. If a custom destructor is necessary, what other functions may also need to be customized?
    (a) Assignment operator
    (b) Default constructor
    (c) Copy constructor
    (d) (a) and (b)
    (e) (a) and (c)
    (f) (b) and (c)
    (g) All of the above.
    (h) None of the above.

18. I created a new class, named `SafeInt` that acts much like an int, but raises exceptions for certain unsafe operations (like division by zero). I've implemented all the needed operators (like `operator/`, but I want to be able to combine `int`s and `SafeInt`s. Which functions need to be implemented so that code like the following will work?
```
SafeInt x = SafeInt(2) / 3;
```
    (a) `SafeInt::SafeInt(const int &)`
    (b) `SafeInt SafeInt::operator/(const int &)`
    (c) `SafeInt to_SafeInt(const int &)`
    (d) (a) or (b) will work.
    (e) (a) or (c) will work.
    (f) (b) or (c) will work.
    (g) All of the above will work.
    (h) None of the above.

19. Which of the following is not an `ostream`?
    (a) `cout`
    (b) `cin`
    (c) `fstream`
    (d) `ofstream`
    (e) `cerr`
    (f) All of the above are `ostream`s.

20. Which of the following operators do all `std::ostream`s support?
    (a) `operator()`
    (b) `operator>>`
    (c) `operator<<`
    (d) `operator[]`
    (e) `operator+`

21. If the following code compiles successfully, which is NOT a possible type for "out_stream"?
```
out_stream << "Hello test taker!";
```
    (a) `std::string`
    (b) `std::stringstream`
    (c) `std::ostream &`
    (d) `std::ofstream`

22. What type must x be for the following code to compile?
```
??? x = (cout << 4)
```
    (a) The type `auto`
    (b) An integer type
    (c) `std::string`
    (d) A reference to a ostream
    (e) No type exists that `x` could be as the code above can't possibly compile
    (f) A different type to any of the above

This page intentionally left blank.