Sample Final Exam
CSE 232 (Introduction to Programming II)
Summer 2019

VERSION 1

Full Name: .................................................................

PID: .................................................................

Instructions:
• DO NOT START/OPEN THE EXAM UNTIL TOLD TO DO SO.
• You may however write and bubble in your name, PID and VERSION/FORM NUMBER (with a #2 pencil) on the front of the printed exam and bubble sheet prior to the exam start.
• Present your MSU ID (or other photo ID) when returning your bubble sheet and printed exam.
• Only choose one option for each question.
• 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, please raise your hand and a proctor will assist you.

http://xkcd.com/499/
1. Should all the questions on the sample (and actual) midterm be included in this sample final?
   (a) Yes, the final is comprehensive
   (b) No, the final is not comprehensive
   (c) No, topics covered on the previous exam will not be on the final.

2. 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

3. When is an array degraded?
   (a) When the scope changes.
   (b) When it is passed to a function.
   (c) When the array is subscripted.
   (d) All of the above.
   (e) When it is used in a loop.

4. Why did we teach arrays in CSE 232?
   (a) Because they are necessary to use pointers.
   (b) Because they represent the simplest compound data structure.
   (c) Because modern C++ makes heavy use of them.
   (d) Because they are an essential data structure.
   (e) None of the above.

5. Which of the following causes memory leaks?
   (a) Allowing variables to fall out of scope.
   (b) Copying pointers and references.
   (c) Not delete'ing new'ed memory.
   (d) Accessing uninitialized memory.
   (e) None of the above.

6. What does the [ ] of a delete [ ] x; indicate?
   (a) That x is a pointer.
   (b) That x is subscriptable.
   (c) That x is a data structure.
   (d) That x is an array.
   (e) None of the above.

7. In the lab where you created your own vector class in the student namespace. What affect would changing the object returned by the member function front have?
   (a) Changes to this object are impossible as front is a const function member.
   (b) Changes to this object are impossible as front returns a const object.
   (c) Changes to this object would alter the vector it belongs to.
   (d) Changes to this object would not alter the vector it belongs to.
   (e) None of the above.

8. What does marking a function member const indicate?
   (a) That it can be invoked on const objects.
   (b) That it doesn't change the object it belongs to.
   (c) That it returns a const object.
   (d) All of the above.
   (e) Only (a) and (b).
   (f) Only (a) and (c).
   (g) Only (b) and (c).
   (h) None of the above.
9. In the lab where you created a SingleLink list, what are the cases that the \texttt{del} function member must account for?

(a) If the argument is the last element in the list.
(b) If list is empty.
(c) If the argument is in the list (but isn’t the first or last element).
(d) If the argument is the first element in the list.
(e) If the argument isn’t present in the list.
(f) All of the above.

10. In the lab where you created a SingleLink list, would a \texttt{out\_of\_range} exception need to be raised for a negative indices?

(a) Yes, because a negative index would access a memory position prior to the first element.
(b) No, because negative indices loop around like in Python.
(c) Yes, because a negative index would result in a memory leak.
(d) No, because negative indices are impossible.
(e) No, because no test case had negative indices.
(f) None of the above.

11. If you expect a function your code will call may raise an exception, which keywords are needed to handle the exception?

(a) \texttt{try} \\
(b) \texttt{finally} \\
(c) \texttt{except} \\
(d) \texttt{throw} \\
(e) \texttt{catch} \\
(f) None of the above.

12. In order to use the assert function, you need to include the \texttt{cassert} header. What does the \texttt{c} in the name mean?

(a) That the header is from the C language.
(b) That the function requires a compiler pre-processor.
(c) That the function runs at compile time.
(d) None of the above.

13. When can you use a \texttt{catch} block without a \texttt{try} block?

(a) When any exception can be caught.
(b) When you want the code in the block to always run.
(c) When it is unclear where an exception can be thrown.
(d) When the statement immediately before it may raise an exception.
(e) None of the above.

14. Which of the following ways to indicate an error state was not demonstrated in this class?

(a) Returning a iterator which points at a data structure’s end.
(b) Throwing an exception.
(c) Returning a \texttt{bool} that indicates success.
(d) Returning a sentinel value.
(e) All of the above were taught.

15. How does the two parameter version of \texttt{stoi} indicate that it was unable to process a string in its entirety?

(a) By returning an iterator and a bool, where the bool indicates success.
(b) By returning a sentinel value.
(c) By adjusting a value pointed at by its second argument.
(d) By throwing an exception.
(e) None of the above.
16. 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).

17. Which of the following types provided by the STL are not templated?

(a) `cout`
(b) `unordered_set`
(c) `multimap`
(d) `vector`
(e) All of the above.
(f) None of the above.

18. 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).

19. When is an object's destructor called?

(a) When the block of that local variable ends.
(b) When the delete operator is called on that object.
(c) When that object is assigned to.
(d) (a) and (b)
(e) (a) and (c)
(f) (b) and (c)
(g) All of the above.
(h) None of the above.

20. 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.

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

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

22. 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.
23. What type can \( x \) be if \( x->\text{begin() \footnotesize is not a type error?} \)

(a) A pointer to a \texttt{vector\textless long\textrangle}.
(b) A pointer to a \texttt{set\textless long\textrangle}.
(c) A iterator of a \texttt{vector\textless vector\textless long\textrangle}.
(d) A iterator of a \texttt{set\textless set\textless long\textrangle}.
(e) (a) and (b)
(f) (a) and (c)
(g) (c) and (d)
(h) (b) and (d)
(i) All of the above.
(j) None of the above.

24. Which of the following iterators can be legally deferred for a vector named \( x \)?

(a) \texttt{x.begin()}
(b) \texttt{x.cbegin()}
(c) \texttt{x.rbegin()}
(d) \texttt{x.crend()}
(e) (a) and (b)
(f) (a), (b) and (c)
(g) All of the above.
(h) None of the above.

25. What is wrong with the following code?
\begin{verbatim}
vector<int> v = {1, 2, 3}
while (v.size()) {
    cout << *v.begin() << endl;
    v.pop_back();
}
\end{verbatim}

(a) It will generate a syntax error because \texttt{*v.begin() you can’t dereference a vector.}
(b) It will print 1 three times because it is popping from the end, not the beginning.
(c) It is an infinite loop because size can never be negative.
(d) It will raise an exception because \texttt{push_back} will be called on an empty vector.

26. What will change if \texttt{struct} is changed to \texttt{class} in the following?

\begin{verbatim}
struct Test {
    double gpa;
    public:
        string name;
    private:
        string comments;
}
\end{verbatim}

(a) The data member \texttt{gpa} will be made private.
(b) Nothing, it still won’t compile because of a missing semicolon.
(c) All three data members will become private.
(d) The data member \texttt{gpa} will be made public.
(e) All three data members will become public.

27. A pointer to an element in an array has the properties of which type of iterator?

(a) Bi-directional iterator
(b) Random access iterator
(c) Input iterator
(d) Forward iterator
(e) Output iterator
(f) All of the above

28. When does the Google Style Guide \textbf{NOT} recommend having a trailing underscore after a data member name?

(a) For data members that are const
(b) For private data members
(c) For data members of structs
(d) For data members that don’t have accessors
(e) It always recommends the underscore
29. What special privileges does a friend function have over a class?
   (a) It can alter const objects.
   (b) It can make public members private.
   (c) It can call deleted functions.
   (d) It can define the operator<<.
   (e) None of the above.

30. Is the following legal?
    template <typename T>
    class Node {
    public:
        T data_;  
    private:
        Node<T> * next_; 
    }; 

    (a) No, it has a syntax error.
    (b) No, a class must have function members.
    (c) No, a type cannot refer to itself.
    (d) No, a data member cannot be a pointer.
    (e) Yes, it is legal.

31. Where should comments explaining how to use a function be located?
   (a) At the top of the header file.
   (b) On the same line as complicated code.
   (c) At the top of the implementation file.
   (d) Immediately before the function implementation.
   (e) Immediately before the function declaration.
   (f) None of the above.

32. 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 ints and SafeInts. 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.

33. Why is self-assignment (i.e. x = x; a concern for types with a non-synthetic operator=?)

    (a) Self-assignment is illegal, and all classes must an raise exception if detected.
    (b) It isn’t a concern, as self-assignment doesn’t have any effect.
    (c) Self-assignment is a syntax error, so classes need not be concerned by it.
    (d) Without care, self-assignment could result in unnecessary work being performed.
34. How many different instances (copies) of vector are generated by the following code?

```cpp
bool has_1(vector<int> input) {
    return find(
        input.begin(),
        input.end(),
        1) != input.end();
}

int main() {
    vector<int> v = {1, 2, 3};
    vector<int> v2 = v;
    cout << has_1(v) << endl;
}
```

(a) 1  (c) 3  (e) 5  
(b) 2  (d) 4  (f) 6

35. Assuming 32-bit ints, how many bytes of memory does the following code leak?

```cpp
for (int i = 0; i <= 3; ++i) {
    int * ptr = new int[i];
}
```

(a) 72  (d) 12  (g) 3  
(b) 48  (e) 9  (h) 0  
(c) 24  (f) 6

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

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

37. What does the following code output?

```cpp
map<int, string> id_to_name = {
    {10, "Josh"},
    {20, "Emily"},
};

id_to_name.insert({30, "Abdol"});
string x = id_to_name[40];
cout << id_to_name.size() << endl;
```

(a) 2  
(b) 3  
(c) 4  
(d) None of the above (it won’t compile)

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

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

39. What does the following code output:

```cpp
vector<int> xs = {1, 2, 3, 4};
for (auto x : xs) {
    xs.push_back(5);
}
cout << xs.size() << endl;
```

(a) 5  
(b) Undefined behavior  
(c) 4  
(d) 9  
(e) Syntax error
40. Can `std::sort` be used to sort an array of floats?

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

41. 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.

42. What will be the output of the following code?

```cpp
void foo(int *p) {
    cout << *p << endl;
    (*p)++;
}

int main() {
    int i = 10;
    foo(&i);
}
```

(a) 11
(b) Segmentation fault/code crash
(c) Compile time error
(d) Run time error
(e) 10
(f) Some garbage value

43. What will be the output of the following code?

```cpp
void foo(int *p) {
    cout << *p << endl;
}

int main() {
    int i = 10, *p = &i;
    foo(p++);
}
```

(a) 11
(b) Segmentation fault/code crash
(c) Compile time error
(d) Run time error
(e) 10
(f) Some garbage value

44. What will be the output of the following code?

```cpp
void foo(int *p) {
    cout << *p << endl;
}

int main() {
    int i = 10, *p = &i;
    foo(++p);
}
```

(a) Compile time error
(b) 10
(c) 11
(d) Some garbage value or error

45. What will be the output of the following code?

```cpp
void foo(int *p) {
    p = new int(2);
    cout << *p << ' ';
}

int main() {
    int i = 97, *p = &i;
    foo(&i);
    cout << *p << endl;
}
```

(a) Run time error
(b) 2 97
(c) Segmentation fault/code crash
(d) Compile time error
(e) 2 2
46. What will be the output of the following code?

```cpp
void foo(int **p) {
    *p = new int(2);
    cout << **p << ' ';
}

int main() {
    int i = 97, *p = &i;
    foo(&p);
    cout << *p << endl;
}
```

(a) Compile time error
(b) 2 97
(c) Segmentation fault/code crash
(d) 2 2
(e) Run time error

47. Which option describes passing a single-dimensional array (of ints) as an argument in a function?

(a) `int sum(int ary[], size_t size)`
(b) `int sum(int *ary, size_t size)`
(c) `int sum(int & ary[size])`
(d) All of the above
(e) Two of the (a), (b), and (c)

48. What will be the output of the following code?

```cpp
int main() {
    int a[3] = {1, 2, 3};
    int *p = a;
    cout << *p << endl;
}
```

(a) Compile time error
(b) Same address is printed twice
(c) undefined
(d) Two different addresses are printed
(e) Run time error

49. What will be the output of the following code?

```cpp
int main() {
    int a[3] = {1, 2, 3};
    int *p = a;
    cout << *(a+1) << '-' << p[1];
}
```

(a) Different addresses are printed
(b) 1-1
(c) Run time error
(d) Compile time error
(e) 2-2

50. Given following Circular Buffer:
   Array Contents: 4, 3, 1, 5, 2
   Head: 3; Tail: 0
   What is new configuration after a pop_front and a push_back 10?

(a) 10, 3, 1, 5, 2
(b) 10, 3, 1, 0, 2
(c) 4, 3, 1, 5, 10
(d) Invalid

51. Which of the following are user-defined datatypes?

(a) `typedef`
(b) `struct`
(c) invalid
(d) `enum`
(e) All of the above

52. Which of the following describe allocating new memory of type long and of size size?

(a) `new long lptr = *long[size];`
(b) `long *lptr = new long[size];`
(c) `new long *lptr = long[size];`
(d) `long lptr = new *long[size];`
(e) `long *lptr = new *long[size];`

53. Where are local variables are allocated?

(a) Stack
(b) Free memory
(c) Permanent storage area
(d) Heap
54. Choose the statement which is **FALSE** with respect to dynamic memory allocation.

(a) Allocated memory can be changed during the run time of the program based on the requirement of the program
(b) Memory is allocated in a less structured area of memory, known as heap
(c) None of the above
(d) Used for unpredictable memory requirements
(e) Dynamically allocated memory is automatically deleted when it falls out of scope.

55. Which of the following is an example for static memory allocation?

(a) Queue
(b) Linked list
(c) Array
(d) None of the above
(e) Stack