CSE 332S: Fall 2021

Name:	
Student ID:	
Date:	

You will have 120 minutes for this exam. **Do not open the exam or start until the designated start time (6:30 PM).** You may begin to look through the provided code if you would like.

As a reminder:

- The exam is closed book and no electronics are allowed
- You are not allowed to collaborate with your neighbors
- You are allowed to use both sides of a single 8.5 inch X 11 inch paper

For this exam, you will use the supplied code to answer questions. The files that should be supplied to you are below. Beside each file name you will find the problems that require it:

File Name	**When to use?**
my_string.h, my_string.cpp	3, 4
palindrome.h, palindrome.cpp	4

Problem #	Points possible	Points earned
1	20	
2	5	
3	22	
4	14	
Extra credit	(3)	
Total	61 (64)	

Problem 1. (20 points) For this problem, you will be given snippets of code and asked questions about those snippets. You should assume we are using namespace std and all necessary header files are included.

Each of the code snippets below contains either a compile time error, a potential runtime error, an unsafe operation, or contains no error and produces output**.** For each code snippet:

- If the code contains an error or unsafe operation, place the letter associated with the error in the table below into the output column for that snippet. Errors may be used more than once. An error does not have to be used. Any output produced before the error occurred can be ignored (or you can include it, but you should clearly mark an error occurred).

- If the code does not contain an error, give its output.

Possible errors:

A. Can't alias a const variable via a reference or pointer to a non-const type	B. illegal/unsafe pointer dereference
C. Illegal pointer arithmetic	D. Uninitialized reference
E. Can't modify a const variable	F. Use of a reference or pointer to an object that has been destroyed
G. Memory leak	H. Double deletion
I. Operator or function is not defined for the given type	J. Indexing out of the bounds of a container, behavior is undefined
K. Exception propagated out of main() uncaught	L. Ambiguous function call

Code:	Output or error:
`void foo(int& i) {` `  int j = 10;` `  i = j;` `  j = 0;` `}` `int & foo(int* p) {` `  int & i = *p;` `  i = 5;` `  return i;` `}` `int main(){` `  int k = 5;` `  foo(k);` `  cout << k << endl;` `  return 0;` `}`	
`int & foo(int* p) {` `  int & i = *p;` `  i = 5;` `  return i;` `}` `int main() {` `  int k = 10;` `  int& j = foo(&k);` `  cout << k << "  " << j << endl;` `  return 0;` `}`	

```
int & foo(int* p) {
    int i = *p;
    i = 5;
    return i;
}

int main() {
    int k = 10;
    int & j = foo(&k);
    cout << k << "  " << j << endl;
    return 0;
}
```

```
void foo(int i, int& j, int* k) {
    ++i;
    k = &j;
    *k = 10;
    k = &i;
    *k = 5;
}


int main()
{
    int i = 10;
    int j = 20;
    int k = 30;
    foo(i, j, &k);
    cout << i << endl;
    cout << j << endl;
    cout << k << endl;
    return 0;
}
```

```
int main()
{
    int arr[] = { 5,10,15,20,25 };
    int* p = arr + 1;
    cout << *p << endl;
    cout << *(arr + 1) << endl;
    cout << arr[1] << endl;
    return 0;
}
```

`` int main() { int arr[] = { 5,10,15,20,25 }; int* p = arr + 1; cout << (arr + 4) - p << endl; cout << *(arr + 4) - *p << endl; return 0; } ```	
`` int main() { int arr[] = { 5,10,15,20,25 }; const int* q = &arr[3]; arr[3] = 50; cout << *q << endl; q = &arr[2]; cout << *q << endl; return 0; } ```	
`` int main() { int arr[] = { 5,10,15,20,25 }; int* const r = &arr[2]; *r = 100; cout << arr[2] << endl; r = &arr[1]; cout << *r << endl; return 0; } ```	
`` int main() { int arr[] = { 5,10,15,20,25 }; int& ref = arr[2]; ref = arr[4]; cout << arr[2] << endl; cout << arr[4] << endl; return 0; } ```	

```
int main()
{
    int arr[] = { 5,10,15,20,25 };
    int& ref;
    ref = arr[4];
    cout << arr[2] << endl;
    cout << arr[4] << endl;
    return 0;
}
```

```
int main()
{
    const char* c = "CSE";
    const char* n = "332S";
    cout << c << n << endl;
    cout << *c << *n << endl;
    return 0;
}
```

```
int main()
{
    const char* c = "CSE";
    const char* n = "332S";
    cout << c << n << endl;
    cout << c + n << endl;
    return 0;
}
```

```
int main()
{
    string cpp_s = "CSE";
    string cpp_n = "332S";
    cout << cpp_s << cpp_n << endl;
    cout << cpp_s + cpp_n << endl;
    return 0;
}
```

`` int main() { const string cpp_s = "CSE"; const string cpp_n = "332S"; cout << cpp_s + cpp_n << endl; return 0; } ```	
`` int main() { const string cpp_s = "CSE"; string& cpp_r = cpp_s; cpp_r[0] = 'E'; cout << cpp_s << endl; return 0; } ```	
`` int main() { const int i = 10; const int* p = &i; cout << *p << endl; delete p; p = nullptr; return 0; } ```	
`` int main() { int i = 10; const int* p = &i; cout << *p << endl; p = nullptr; return 0; } ```	

```
int main()
{
    int i = 10;
    int* p = new int(i);
    cout << *p << endl;
    p = &i;
    cout << *p << endl;
    return 0;
}
```

```
int main()
{
    int i = 10;
    int* p;
    cout << *p << endl;
    return 0;
}
```

```
int main()
{
    vector<int> v;
    v[0] = 1;
    v[1] = 2;
    int& p = v[1];
    cout << p << endl;
    return 0;
}
```

Problem 2. (5 points) Given the following description of a potential program I might implement, answer the questions.

Description: The program will create several objects of a class called *Foo* and insert those objects into a sequential container. The program will then sort the objects using std::sort() and write each of the sorted objects to a file via an output file stream.

Questions:

A. What operators, functions, or constructors should be implemented for the Foo class?

B. Which sequential containers may be used to store the Foo objects?

Problem 3. (22 points) Using the code provided (my_string.h, my_string.cpp) and the main function below, answer the following questions.

- Assume we are **using the std namespace** and **all necessary header files are included.**
- For each question that asks for the output at a certain line number**,** assume the code has executed correctly up to that point and **give the output produced by that line of code only.** Do not give the output of the entire program. (Note all constructors and destructors have *cout* statements in them).

```
int main()
{
    const char* h = "hello";
    my_string s(h, 5);                  // 4
    s.print();                          // 5
    my_string s2(s);                    // 6
    s2[0] = 'm';
    s.print();                          // 8
    s = s2;                             // 9
    s.print();                          // 10
    s2[0] = 'y';
    // cout << s[5] << endl;            // 12
    s = s + s2;                         // 13
    s.print();                          // 14
    return 0;
}
```

A. (5 points) List the 5 basic copy control operations. Beside each operation, list the line number in *my_string.h* where the operation is declared for the my_string class.

B. (1 point) The copy constructor of the my_string class makes a _____ copy. Circle one:

Shallow Deep

C. (1 point) The copy assignment operator makes a _____ copy. Circle one:

Shallow Deep

D. (1 point) What is the output produced by line number “4”? (marked “4” in a comment)

E. (1 point) What is the output produced by line number “5”? (marked “5” in a comment)

F. (1 point) What is the output produced by line number “6”? (marked “6” in a comment)

G. (1 point) What is the output produced by line number “8”? (marked “8” in a comment)

H. (1 point) What is the output produced by line number “9”? (marked “9” in a comment)

I. (1 point) What is the output produced by line number “10”? (marked “10” in a comment)

J. (1 point) Line “12” (marked “12” in a comment) has to be commented out or it will cause an error. Describe the error it would cause.

K. (2 points) What is the output produced by line number “13”? (marked “13” in a comment)

L. (1 point) What is the output produced by line number “14”? (marked “14” in a comment)

M. (1 point) When the main function returns, what output is produced?

N. (2 points) Currently, this program has no memory errors. If you notice one, point it out for extra credit. If **line 53 of my_string.cpp** were commented out, what memory error would be introduced into this program. Describe the error.

O. (2 points) If **line 69 in my_string.cpp** were commented out, what memory error would be introduced into this program. Describe the error.

Problem 4. (14 points) Using the code provided (my_string.h, my_string.cpp, palindrome.h, palindrome.cpp) and the main function below, answer the following questions.

- Assume we are **using the std namespace** and **all necessary header files are included.**
- For each question that asks for the output at a certain line number**,** assume the code has executed correctly up to that point and **give the output produced by that line of code only.** Do not give the output of the entire program. (Note all constructors and destructors have *cout* statements in them).
- For anyone unfamiliar, a palindrome is a word that reads the same forwards and backwards. In this implementation, I only store half the characters in a palindrome to save memory.

```
int main()
{
    palindrome p("anna", 4);          // 3
    my_string& r = p;
    cout << r[3] << endl;             // 5
    r.print();                        // 6
    my_string s(p);                   // 7
    my_string* sp = &s;
    sp->print();                      // 9
    return 0;
}
```

A. (2 points) What is the output produced when line number 3 executes (the line marked 3 by a comment)?

B. (1 point) As far as the compiler is concerned, what type of object does “r” refer to (static type)?

C. (1 point) At run-time, what type of object does “r” refer to (dynamic type)?

D. (1 point) What is the output produced by line number 5 (marked by “5” in a comment)?

E. (2 points) What is the output produced by line number 6 (marked by “6” in a comment)?

F. (1 point) What is the output produced by line number 7 (marked by “7” in a comment)?

G. (1 point) As far as the compiler is concerned, what type of object does “sp” point to (static type)?

H. (1 point) At run-time, what type of object does “sp” point to (dynamic type)?

I. (2 points) What is the output produced by line number 9 (marked by “9” in a comment)?

J. (2 points) When the main function returns, what output is produced?

5. **Extra Credit:** (3 points) Define the += operator for the my_string class. This operator concatenates the string passed as a parameter onto the original string and returns a reference to the string being assigned to. Be sure to avoid any memory related issues.

```
my_string & my_string::operator+=(const my_string & s) {

}
```