INSTRUCTIONS:

(1) DO NOT OPEN YOUR EXAM BOOKLET UNTIL YOU HAVE BEEN TOLD TO BEGIN.

(2) The total for the exam is 100 points

(3) There are 10 pages with 32 problem; 15 multiple-choice, 15 short answer, and 2 free response.

(4) You may skip one multiple-choice and one short answer problem for a total of 5 points.

(5) Use the backside of the paper as your scratch paper. The backside is not graded.

(6) If you make a mistake, cross it out or erase it. Otherwise, it will be graded, for better or for worse.

(7) No electronic devices (e.g. phones, calculators, mp3 players, etc.) are allowed.

(8) You are allowed one sheet of notes on Letter paper or smaller.

(9) Write your name on the upper-right corner of each page of the exam.

(10) To receive full credit, you must write legibly and your answers must be clear. You may need to sketch out answers to less straightforward problems on the scratch side before you write on the front side.

(11) Do not leave classroom until you have handed in the exam.

(12) The exam lasts 110 minutes
(1) (2 pts) Which of the following describes the term “memory leak”?
   (a) An undeclared array
   (b) A pointer that points to the value nullptr
   (c) Memory that was allocated by new but not deleted with delete
   (d) Memory that is untyped
   (e) None of the above

(2) (2 pts) What is the Big-Oh rating for the algorithm to search for an element in a singly-linked list?
   (a) $O(1)$
   (b) $O(n \cdot \log(n))$
   (c) $O(n)$
   (d) $O(n^2)$
   (e) None of the above

(3) (2 pts) What is the return type of a class constructor?
   (a) An instance of the class
   (b) No return on a constructor
   (c) An int (the size of the class in bytes)
   (d) A string (the name of the class)
   (e) None of the above

(4) (2 pts) Which of the following represent differences between a C++ class and a struct?
   (a) struct cannot have associated methods
   (b) Only a class can have private data members
   (c) There are no constructors or destructors for a struct
   (d) All of the above
   (e) None of the above

(5) (2 pts) What of the following are true about the special variable this?
   (a) It is a reference value
   (b) You are allowed to change what it points to in a method
   (c) It is bound to the calling instance of a method
   (d) All of the above
   (e) None of the above
(6) What is the size of a C++ pointer (how many bytes)?
   (a) Depends on the size of addressable memory of the operating system (32 bit, 64 bit, etc.)
   (b) Same size as the type it points to
   (c) Always 2 bytes
   (d) Always 4 bytes
   (e) None of the above

(7) What is the Big-Oh rating for selection sort?
   (a) $O(1)$
   (b) $O(n)$
   (c) $O(n \cdot \log n)$
   (d) $O(n^2)$
   (e) None of the above

(8) What is the “rule of three” for C++ classes?
   (a) If you provide a copy constructor, destructor, or operator=, you should provide them all
   (b) You must provide three constructors in a dynamic memory class
   (c) There should be no more than three private variables in a class
   (d) You need a destructor if the size of a private data member dynamic array is bigger than 3
   (e) None of the above

(9) Which of the following are true about STL vectors and regular C++ arrays?
   (a) Both have their first value at index 0
   (b) Both can use generic algorithms by using STL iterators
   (c) Both are fixed in their size.
   (d) All of the above
   (e) None of the above
template<typename T>
T fn1(const vector<T>& p1, const T p2) {
    T result = T(); // Initialize to default T
    for (auto e : p1) { // Line 1
        if (e > p2)
            result += e;
    }
    return result;
}

int main() {
    vector<long> v_l = {1, 2, 3, 1, 2, 3};
    long long_arg = 2;
    cout << fn1(v_l, long_arg) << endl; // Line 2
    vector<string> v_s = {"ab", "cd", "ef", "ab", "cd", "ef"};
    string str_arg = "ab";
    cout << fn1(v_s, str_arg) << endl; // Line 3
}

(10) What is the type of e on Line 1 given the call on Line 2?
    (a) long
    (b) long*
    (c) vector<long>::iterator
    (d) vector<long>
    (e) None of the above

(11) What value is output on Line 2?
    (a) 3
    (b) 4
    (c) 5
    (d) 6
    (e) None of the above

(12) What value is output on Line 3?
    (a) ab
    (b) abab
    (c) cdef
    (d) The empty string
    (e) None of the above
```
template<typename T>
int fn2(vector<T>& v, T val) {
    auto res1 = find(v.begin(), v.end(), val); // Line 1
    auto res2 = find(res1 + 1, v.end(), val);
    if (res2 != v.end()) {
        return res2 - v.begin();
    } else {
        return -1;
    }
}

int main() {
    vector<long> v_l = {1, 2, 3, 1, 2, 3};
    long long_arg = 2;
    cout << fn2(v_l, long_arg) << endl; // Line 2
    vector<string> v_s = {"ab", "cd", "ef", "ab", "cd", "ef"};
    string str_arg = "ab";
    cout << fn2(v_s, str_arg) << endl; // Line 3
}
```

(13) What is the type of `res1` on Line 1 given the call on Line 3?

(a) char  
(b) string  
(c) vector*  
(d) string*  
(e) None of the above

(14) What value is output on Line 2?

(a) 1  
(b) 2  
(c) 3  
(d) 4  
(e) None of the above

(15) What value is output on Line 4?

(a) 1  
(b) 2  
(c) 3  
(d) 4  
(e) None of the above
```cpp
struct MyStruct {
    long l1 = 0;
    long l2 = 0;
    MyStruct() = default;
    MyStruct(long p1, long p2): l1(p1), l2(p2) {};
};

(16) What is the output of MyStruct s; cout << s.l1 << endl;? 0

(17) What is the output of MyStruct s(2, 3); cout << s.l1 << endl;? 2

void fn3(MyStruct& s, long lng) {
    s.l1 += lng;
    s.l2 += lng;
}

(18) After executing MyStruct s; fn3(s, 1);, what is the value of s.l1? 1

(19) After executing MyStruct s(2, 3); fn3(s, 2);, what is the value of s.l1? 4

bool fn4(const MyStruct& s1, const MyStruct& s2) {
    auto res1 = static_cast<double>(s1.l1) / s1.l2; // Line 1
    auto res2 = static_cast<double>(s2.l1) / s2.l2;
    return res1 < res2;
}

int main() {
    vector<MyStruct> v{ {1, 2}, {3, 3}, {4, 5}};
    sort(v.begin(), v.end(), fn4);
    cout << v[0].l2 << endl; // Line 2
    cout << v[1].l2 << endl; // Line 3
}

(20) What is the return type of res1 on Line1? double

(21) What is the result of Line 2? 2

(22) What is the result of Line 3? 5
class MyClass {
public: // Everything public for easier testing
    char *c_ = nullptr;
    size_t sz_ = 0;
    size_t mx = 0;
    MyClass() = default;
    MyClass(initializer_list<char>);
    int method1(char);
    size_t method2(char);
};
MyClass::MyClass(initializer_list<char> i_c) {
    sz_ = mx_ = i_c.size();
    c_ = new char[mx_];
    copy(i_c.begin(), i_c.end(), c_);
}
int MyClass::method1(char c) {
    auto res = find(c_, c_+sz, c);
    if (res == c_+sz)
        return -1;
    else
        return res - c_;
}
size_t MyClass::method2(char c) {
    if (sz_ == mx_){
        mx_ *= 2;
        char* temp = new char[mx_];
        copy(c_, c_+sz, temp);
        swap(temp, c_);
        delete[] temp;
    }
    c_[sz_++] = c;
    return sz_;}
int main() {
    MyClass mc = {'a', 'b', 'c'};
    cout << mc.sz_ << endl; // Line 1
    cout << mc.mx_ << endl; // Line 2
    cout << mc.method1('d') << endl; // Line 3
    cout << mc.method1('c') << endl; // Line 4
    cout << mc.method2('d') << endl; // Line 5
    cout << mc.method1('d') << endl; // Line 6
    cout << mc.sz_ << endl; // Line 7
```
cout << mc.mx_ << endl; // Line 8
```

(23) What is the output produced on Line 1? 3

(24) What is the output produced on Line 2? 3

(25) What is the output produced on Line 3? -1

(26) What is the output produced on Line 4? 2

(27) What is the output produced on Line 5? 4

(28) What is the output produced on Line 6? 3

(29) What is the output produced on Line 7? 4

(30) What is the output produced on Line 8? 6
(31) (15 pts) A (math) vector in polar coordinates is represented by a radius $r$ and an angle $\theta$. Write a `Polar` class that represents a polar vector. Additionally, you must write a * operator that multiplies two `Polar` objects together to compute a new `Polar` object representing their product. Hint: when multiplying two polar vectors, you multiply their radii and add their angles.

```cpp
struct Polar {
    double radius;
    double angle;

    Polar(double r = 0, double a = 0) :
        radius(r), angle(a) {} 
};

Polar operator*(const Polar& x, const Polar& y) {
    return Polar(x.radius * y.radius, x.angle + y.angle);
}
```
(32) (15 pts) Write a function `BuildSumArray` that takes two arrays of length $n$ (another parameter) and creates and returns a new array of length $n$ that contains the sums of the corresponding elements in the first two arrays. **Your function must be templated to work on many types.**

```cpp
template <class T>
T* BuildSumArray(T* A, T* B, size_t n) {
    T* C = new T[n];
    for (size_t i = 0; i < n; i++) {
        C[i] = A[i] + B[i];
    }
    return C;
}
```