**CS A250 – Practice Final**

**Question 1.** What is the output of the following code segment? **5 4 7 6 7 7**

```
int a = 3, b = 1, c = 2, d = 6;
do
{
        a += b;
        c = (++a < b) ? b : a;
        --c;
        cout << a << c;

} while (c++ < d);
cout << a << c;
```

**Question 2.** What is the output of the following code segment? **3 3 4 3 2**

```
void oneParam(int& c)
{
        ++c;
        cout << c;
}
void twoParam(int& a, int b)
{
        a -= b;
        b++;
        cout << a << b;
        oneParam(b);
}
int main()
{
        int a = 5, b = 2;
        twoParam(a, b);
        cout << a << b;
        return 0;
}
```

**Question 3.** What is the output of the following code segment? **8 8 8**

```
int a = 3, b = 5;
int *p1 = new int;
int *p2 = new int;
*p1 = a;
*p2 = a + b;
int *p3 = p2;
p1 = p3;
cout << *p1 << *p2 << *p3;
```

**Question 4.** Given the array below, how many comparisons will be performed by an iterative binary search algorithm (as seen on the slides) if searching for 60? **4**

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | 15 | 18 | 29 | 32 | 35 | 41 | 49 | 56 | 57 | 59 | 60 | 61 | 63 | 67 | 70 | 71 | 72 | 75 | 77 | 83 | 84 | 88 | 89 | 94 | 97 | 99 |

There are 27 elements.
Middle element = (0 + 26) / 2 = 13 (index) => **compared** => not equal

| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 12 | 15 | 18 | 29 | 32 | 35 | 41 | 49 | 56 | 57 | 59 | 60 | 61 |

There are 13 elements.
Middle element = (0 + 12) / 2 = 6 (index) => **compared** => not equal

| 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|
| 49 | 56 | 57 | 59 | 60 | 61 |

There are 6 elements.
Middle element = (7 + 12) / 2 = 19 / 2 = 9 (index) => **compared** => not equal

| 10 | 11 | 12 |
|---|---|---|
| 59 | 60 | 61 |

There are 3 elements.
Middle element = (10 + 12) / 2 = 22 / 2 = 11 (index) => **compared** => equal

**Question 5.** What is the running time (**O-notation**) of the **destroyList** function implemented in the **singly-linked list** class? **O(n)**

**Question 6.** What is the running time of the code segment shown below? **O (n^2)**

```
for (int i = length - 1; i >= 0; --i)
{
        for (int j = 0; j < length / 2; ++j)
        {
                int k = 8;
                while (k > 0)                   // the loop will iterate independent from n
                {
                        cout << (length / 2);
                        cout << a[k] << " ";
                        k /= 2;
                }
        }
}
```

Figures 1-3 show nodes that are part of a **doubly-linked list** where **ptr** is a pointer pointing at one of the nodes. Using the functions below and referring to the appropriate figures, answer questions 1-3.

```
Node* getNext() const;          // returns the address of next node
Node* getPrevious() const;      // returns the address of previous node
void setNext(Node *p);          // re-sets the link that points to next node
void setPrevious(Node *p);      // re-sets the link that points to previous node
```

Fig. 1

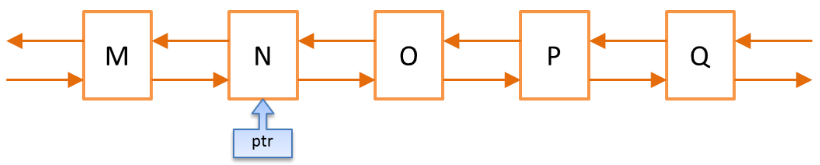


Fig. 2

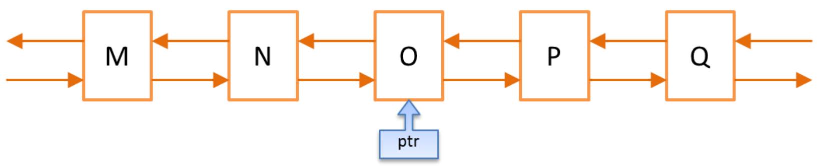
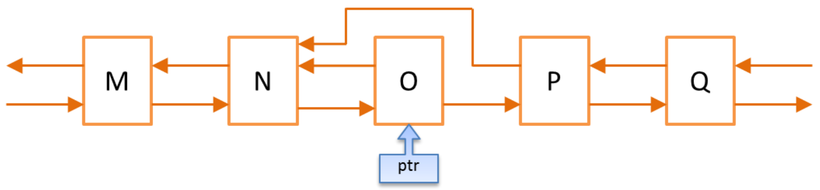


Fig. 3



**Question 7.** In **Fig. 1**, point **ptr** is pointing to **node N**. Write **one (1) single statement** to move **ptr** so that it points to **node O**, as shown in **Fig. 2**.

```
ptr = ptr->getNext ();
```

**Question 8.** Write **one (1) single statement** to change the structure of the doubly-linked list from **figure 2** to **figure 3**, **without** moving the pointer. The new list will have the **previous** link of **node P** point **to node N**.

```
ptr->getNext ()->setPrevious(ptr->getPrevious());
```

**Question 9.** Write **one (1) single statement** to delete **node N** in **figure 3**, **without** moving the pointer. Do not worry about re-connecting the nodes.

```
delete ptr->getPrevious();
```

**Question 10.** What is the output of this code segment? **7**

```
void check(vector<int> v, int& sum)
{
        int size = static_cast<int>(v.size());
        for (int i = 1; i < size; ++i) sum += v[i];
}
int main()
{
        vector<int> v;
        int sum = 0;
        for (int cnt = 3; cnt >= 1; --cnt) v.push_back(cnt + 1);
        check(v, sum);
        sum += v[2];
        cout << sum;
        return 0;
}
```

**Question 11.** What is the output of this code segment? **5 4 5 4 2 5**

```
vector<int> v = { 4, 9, 7, 5, 4, 2, 5, 4, 7, 2, 5, 9, 7 };
vector<int>::iterator iter = v.begin() + 1;
int i = 0;
while (i < 3)
{
        ++iter;
        cout << iter[1];
        cout << *(iter + 2);
        iter += 2;
        ++i;
}
```

**Question 12.** What is the output of the following code segment? **5**

```
list<int> myList, yourList;
myList.push_back(5);
myList.push_back(6);
myList.push_back(2);
myList.push_front(1);
myList.push_front(7);
myList.push_front(4);
list<int>::iterator iter1 = myList.begin();
list<int>::iterator iter2 = myList.begin();
++iter2;
++iter2;
myList.pop_back();
yourList.push_back(*iter2);
list<int>::iterator iter3 = yourList.begin();
++iter2;
myList.push_front(9);
++iter2;
yourList.push_front(*iter2);

        myList: 9 4 7 1 5 6
        yourList: 6 1
        iter1:  4
        iter2: 6
        iter3: 1
```

**Question 13.** What is the the output of the following code segment? **53**

```
multimap<int, int> myMap;
for (int i = 1; i < 8; i += 2)
        myMap.insert(make_pair(i, i + 2));
for (int i = 2; i <= 16; i *= 2)
        myMap.insert(make_pair(i + 1, i * 2));
multimap<int, int>::iterator iter = myMap.begin();
++iter;
cout << iter->second << iter->first << endl;
```

For questions 14 and 15, refer to the definitions shown below.

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

**Question 14.** Given the definitions shown above, what is the output of the following code segment? **'((3) 6)**

```
(cond
    ((equal? a (first (rest f))) (cons d (rest e)))
    (else (cons (equal? (first f) b) c)))
```

**Question 15.** Given the definitions shown above, what is the output of the following code segment? **'(#t)**

```
(cond
    ((equal? a (first (rest e))) (cons d (rest e)))
    (else (cons (equal? (first f) b) c)))
```

**Question 16.** Given the function shown below, what will the call `(recur 3 ‘(2 3))` return? **4**

```
(define recur
  (lambda (a r)
    (cond
      ((empty? r) 1)
      ((equal? a (first r)) (+ (recur a (rest r)) a))
      (else (recur a (rest r))))))
```

**Question 17.** Define a **DrRacket** function **smallest** that takes two numbers, **x** and **y**, and outputs the smallest of the two. In the case the numbers are the same, it output the word **‘same** (Note that ‘same should have an apostrophe, and not quotes). Your function should work with the following test cases:

```
> (smallest 1 2)        => 1
> (smallest 2 1)        => 1
> (smallest 1 1)        => 'same
```

```
(define smallest
  (lambda (x y)
    (cond
      ((< x y) x)
      ((> x y) y)
      (else 'same))))
```

**Question 18.** Write the declaration of a friend function **modify** of a class **MyClass** that passes a list of integers and a vector of integers and returns a Boolean value. The function compares the list and the vector and removes all the elements of the list that are equal to any of the elements in the vector.

```
friend bool modify(list<int>&, const vector<int>&);
```

// If you wrote the definition, you are not reading the question correctly.