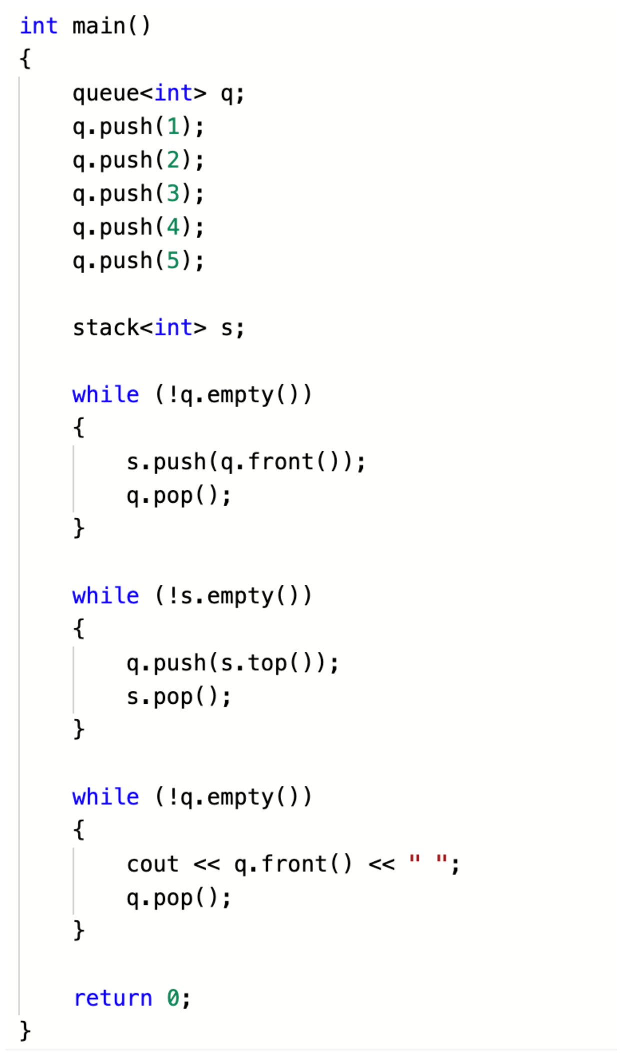
Workshop Review

Please do **not** distribute with others outside the workshop!

1. What's the output?

```
bool isPrime(int num)
{
    if (num <= 1)
        return false;

    for (int i = 2; i < num; i++)
    {
        if (num % i == 0)
            return false;
    }

    return true;
}

int main()
{
    queue<int> q;
    vector<int> v = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

    for (int i = 0; i < v.size(); i++)
    {
        if (isPrime(v[i] + 2))
            q.push(v[i]);
    }

    while (!q.empty())
    {
        cout << q.front() << " ";
        q.pop();
    }
    return 0;
}
```

2. What's the output?



```
int main()
{
    queue<int> q;
    q.push(1);
    q.push(2);
    q.push(3);
    q.push(4);
    q.push(5);

    stack<int> s;

    while (!q.empty())
    {
        s.push(q.front());
        q.pop();
    }

    while (!s.empty())
    {
        q.push(s.top());
        s.pop();
    }

    while (!q.empty())
    {
        cout << q.front() << " ";
        q.pop();
    }

    return 0;
}
```

3. Given vector< vector<int> > numberLists, write a function that returns the sum of all the numbers in numberLists. Note that this is a vector of vectors.

   Make sure to test your program. Create a vector of vectors and pass it to your function. For example:

   Input: < <1,2,3>, <5,5,5>, <2, 0, 1> >
   Output: 24

4. Go to Teams general -> files -> week 7 and do the problem we did a while back named "find kth smallest distance". If you don't know how to use std::pair (though I'd recommend you learn it), you can make a struct to represent a coordinate and then make a vector of that struct to hold all of the coordinates. Please import queue and use the built-in priority_queue to solve this as I stated when we originally assigned this problem.

   Make sure to test your program by creating a vector or array of coordinates. For example:

   Input: <<4,6>, <2,3>, <1,1>, <1,2>> with k = 3
   Output: <2,3> because the coordinate <2,3> is the 3rd smallest distance from the origin.

5. Given a vector < vector<int> > sortedLists, which holds k sorted lists, write a function to merge the k sorted list into one single list. Return the final list as a vector. Don't overthink this problem, it's like the problem we did in the beginning of workshop "merge two sorted linked list", but I've made it easier by using vectors. Hint: Use a priority queue.

   Make sure to test your program by creating a vector of vectors, where each of the vectors inside is sorted. For example:

   Input: < <5,6,7>, <20,21,22>, <1, 2, 3>, <7, 8, 9> >
   Output: <1, 2, 3, 5, 6, 7, 7, 8, 9, 20, 21, 22>

6. *Write a HashTable class that implements hashing using a method of your choice. For example, linear probing, quadratic, etc.*

   I should be able to do something like the following:

```
vector<int> numbers = {1,2,3,4,5,6,7}
int tableSize = 8;
HashTable ht(tableSize);

for (auto num: numbers)
{
    ht.insert(num) // insert should handle collisions
}

bool found = ht.find(5); // find returns true if found false otherwise.
```

7. What's the average and worst look up time in a hash table, and when do these occur?

8. Explain linear probing, quadratic, double, and chaining. How do each of these work and what are the advantages / disadvantages?

9. What's the average and worst time complexity for searching in a binary search tree, and when do these time complexities occur?

10. Describe how each of the following tree traversals work:

    Pre order
    In order
    Post order
    Level order

    Also write the code for each of these

11. What happened if we do an in order traversal on a binary search tree?

    Create a binary search tree and trace this out using the "finger method" short cut I showed y'all.

12. How does inserting into a binary search tree work?

    Also write the code for this

13. How does deleting from a binary search tree work?

    Hint: there are three cases. Start by describing those 3.