**Write a class template for queue class. Assume the programmer using the queue won't make mistakes, like exceeding the capacity of the queue, or trying to remove an item when the queue is empty. Define several queues of different data types and insert and remove data from them.**

```
#include <iostream>
#define SUCCESS 0
using namespace std;

template <typename T>
class Queue
{
 private:
  T data[100];
  int pos;
 public:
  Queue()
  {
    pos = 0;
    for(int i = 0; i< 100; i++)
      data[i]=0;
  }
  void add(T d)
  {
    data[pos] = d;
    pos++;

  }
  T get()
  {
    T  d = data[0];
    for(int i = 0 ; i < pos; i++)
    {
      data[i] = data[i+1];

    }
    pos--;
    return d;
  }
};
```

```
int main()
{
  Queue<int> intlist;
  intlist.add(3);
  intlist.add(4);
  cout << intlist.get() << endl;
  cout << intlist.get() << endl;
  Queue<float> flist;
  flist.add(3.0);
  flist.add(4.9);
  cout << flist.get() << endl;
  cout << flist.get() << endl;
  return SUCCESS;
}
```