CS117 (Fall 2019) Lab Assignment (10 pts) Practice for inheritance: base/derived classes in C++

Study the following C++ program, type the code, compile/run it on your system, and complete the answer sheet (attached) and submit it within the lab session.

```
//stack and queue are implemented with circular lists.
#include <iostream>
using namespace std;

class cell
{ int info; cell *next;
  cell(int i) {info = i; next = this;}
  cell(int i, cell *n) {info = i; next = n;}
  friend class list;
};
class list
{ cell *rear;
  public:
    list() {rear = new cell(0);}
    ~list() {while (!empty()) get();}
    int empty() {return rear == rear->next;}
  protected:
    void add(int);
    void push(int);
    int get();
};

void list::push(int x)
{ rear->next = new cell(x, rear->next); }

void list::add(int x)
{ rear->info = x;
  rear = rear->next = new cell(0, rear->next);
}

int list::get()
{ if (empty()) return 0;
  cell *front = rear->next;
  rear->next = front->next;
  int x = front->info;
  delete front;
  return x;
}

class queue: public list  //derived class
{ public:
    queue() { }
    int Qget() {return list::get();}
    void Qput(int x) {add(x);}
};

class stack: private list //derived class
{ public:
    stack() { }
    int Spop() {return get();}
    void Spush(int x) {list::push(x);}
    using list::empty;  //make inherited member empty public
};

int main()
{ queue q1; stack s1;
  q1.Qput(3); q1.Qput(5); q1.Qput(7);
  cout<<q1.Qget()<<endl; cout<<q1.Qget()<<endl;
  cout<<q1.Qget()<<endl; cout<<q1.Qget()<<endl;

  s1.Spush(2); s1.Spush(4); s1.Spush(6);
  cout<<s1.Spop()<<endl; cout<<s1.Spop()<<endl;
  cout<<s1.Spop()<<endl; cout<<s1.Spop()<<endl;

  return 0;
}
```

Lab Assignment (OOP) answer sheet (10 pts) **Name:**

1. For class queue,
   (a) List public/protected/private members of class queue.
   (b) Is it possible for Qget( ) to access rear ? (circle: Yes, No)
   Is it possible for get( ) to access rear ? (circle: Yes, No)

Public: Protected: Private:

2. For class stack,
   (a) List public/protected/private members of class stack.
   (b) Is it possible for Spush( ) to access rear ? (circle: Yes, No)
   Is it possible for get( ) to access rear ? (circle: Yes, No)
   Is it possible for empty( ) to access rear ? (circle: Yes, No)

Public: Protected: Private:

3. Now, assume that we include the following three statements in the main function:
   (a) cout<<q1.empty();
   (b) cout<<s1.empty();
   (c) q1.list();

   Answer for each statement: Does it cause an error? Justify your answer (explain the reason briefly).

   (a) circle: (error, no_error); reason:

   (b) circle: (error, no_error); reason:

   (c) circle: (error, no_error); reason: