Recursion

Example 1:
```
// Return the factorial of n using recursion
// Assume n is a nonnegative integer
int fact(int n)
{







}
```

Example 2:
```
// Print the elements of an array in order
void printArrayInOrder(int arr[], int n)
{







}
```

How to print the array in reverse order?

Example 3: a = {4, 3, 5, 2, 1, 8, 7, 6}
```
void merge(int a[], int b, int mid, int e)
{
// Assume we can implement this function such that we can merge 2
sorted passes within e computations
// e.g. int a[] = {1, 3, 5, 2, 4, 6}, merge(a, 0, 3, 6), we get a[] =
{1, 2, 3, 4, 5, 6}
}

// Sort the array elements a[b] through a[e−1]
void MergeSort(int a[], int b, int e)
{
    if (e − b > 1){
        int mid = (b+e)/2;
        MergeSort(a, b, mid);
        MergeSort(a, mid, e);
        merge(a, b, mid, e);
    }
}
```

Example 4:
```
// Return a^b
// Assume b is a nonnegative integer
int expon(int a, int b)
{

}
```

Example 5:
```
// Return fibonacci(n)
int fab(int n)
{

}
```

Example 6:
```
// You can go either 1 or 2 steps each time.
// How many ways are there for you to go n steps?
int step(int n)
{

}
```

Problem 7: Parade organization

```
//You are asked to organize a parade consisting of bands and floats.
//You can't place a band immediately after another.
//How many ways can you organize a parade of size n?
//i.e. for n = 2, you have 3 ways to organize the parade: float-float,
float-band and band-float.

void solveParade(int n)
{







}
```

Inheritance

Problem 1: What is the output for the following program?

```
#include <iostream>
using namespace std;
class A
{
public:
	void print(){cout << "A::print()"<<endl;}
};

class B : public A
{
public:
	void print(){cout << "B::print()"<<endl;}
};

int main()
{
	A a;
	a.print();
	B b;
	b.print();
	A* c = new B;//Polymorphism
	c->print();
}
```

Problem 2: What if we add virtual keyword to print() in class A?

Problem 3: What's the output?

```
#include <iostream>
using namespace std;
class A
{
public:
    virtual void print(){cout << "A::print()"<<endl;}
    virtual ~A(){};
};

class B:public A
{
public:
    void print(){cout << "B::print()"<<endl;}
};

void printSomething_1(A a){ a.print();}
void printSomething_2(A& a){ a.print();}
void printSomething_3(A* a){ a->print();}

int main() {
    B b;
    A* c = new B;
    printSomething_1(b);
    printSomething_2(b);
    printSomething_3(&b);
    printSomething_1(*c);
    printSomething_2(*c);
    printSomething_3(c);
}
```