Solutions to Stack and Queue Exercises

1. Write a program that reads in numbers from the standard input until end–of–file, then prints the numbers in reverse order.

```
void
print_reversed(void)
{
  stack_type stack;
  int num;

  stack = create_stack();
  while (scanf("%d", &num) != EOF)
    push(num, stack);
  while (!empty_stack(stack))
    printf("%d\n", pop(stack));
}
```

2. Write the function `copy_queue` that takes a `queue_type` as its argument and returns another `queue_type` as its result. The return queue must contain all of the items in the original queue in the same order that they were found in the original queue. The original queue must end up unchanged. Your function must lie above the queue abstraction barrier (*i.e.*, all manipulation of queues must be done using `create_queue`, `enqueue`, `dequeue`, and `empty_queue`).

```
queue_type
copy_queue(queue_type queue)
{
  queue_type copy;
  int count;
  int i;

  copy = create_queue();
  /* First count the number of items */
  while (!empty_queue(queue)) {
    count++;
    enqueue(dequeue(queue), copy);
  }
  /* Now place that number of items in both queues */
  for (i = 0; i < count; i++) {
    queue_item_type item;

    item = dequeue(copy);
    enqueue(item, queue);
    enqueue(item, copy);
  }
  return(copy);
}
```