PIC 10A 1C Week 8b Problems. TA: Eric Kim *[v2, Updated: 2/25/2016]*

1. Timey-Wimey

Recall the following documentation for the time() function (defined in <ctime>):

```
/**
 * This function takes in a pointer to a time_t variable, and assigns to that
 * variable the number of seconds since Jan 1, 1970 00:00 UTC, then returns
 * this value as a time_t variable. Given a nullptr input, it simply returns
 * the number of seconds since Jan 1st, 1970.
 * @param timeptr the pointer to time_t
 * @return the number of seconds elapsed since Jan. 1st, 1970.
 **/
time_t time(time_t *timeptr);
```

Louis Reasoner wants to write a program that times the number of seconds between user inputs. For instance, the program should behave as follows:

```
    Press enter to start the timer!
    <you pressed enter!>
    Press enter again to stop the timer…
    <you pressed enter!>
    Time elapsed: 12 seconds.
```

Does the following program behave correctly? If not, explain why.

```
#include <ctime>
#include <iostream>
using namespace std;
int main() {
  time_t t;
  cout << "ENTER to start the timer!" << endl;

  cin.get(); // Wait for user to hit enter

  cout << "<you pressed enter!>" << endl;

  time(&t); // Initialize timer t

  cout << "Press enter again to stop the timer...\n";

  cin.get(); // Wait for user to hit enter again

  cout << "<you pressed enter!>" << endl;

  time_t duration = time(nullptr) - t;

  cout << "Time elapsed: " << duration << " seconds.\n";
  return 0;
}
```

2. Hole in my Vector

Write a function `vecremove(vector<int>&vec, size_t ind)` that removes the element at index ind. Make sure that the elements to the right of "ind" are shifted over. If the input index is out of bounds, then the function should not modify the vector - it must not crash:

```
    vector<int> v = {1, 2, 3, 4};
    vecremove(v, 1); // v is now: [1, 3, 4]
    vecremove(v, 0); // v is now: [3, 4]
    vecremove(v,2); // v is still: [3,4]
```

You may not use `vector::erase`.

```
void vecremove(vector<int>& vec, size_t ind) {
    // YOUR CODE HERE
```

3. There Is No Compiler...

For the following, write down the expected output. If an error occurs, describe why. Assume all headers have been included, and that we are using the standard namespace:

`int d = 42;`
`int* dp = &d;`
`*dp = 43;`
`cout << *dp << " " << d;`	

`double x = 42.4;`
`(*x) = (*x) + 1;`
`cout << x;`	

`double x = 13.42;`
`double &xr = x;`
`xr = xr + 1;`
`cout << xr << " " << x;`	

`double x = 1.1;  double y = 3.2;`
`double &xr = x;`
`xr = &y;`
`xr += 1;`
`cout << xr << " " << y;`	

`int x = 1;  int y = 3;`
`int *xr = &x;`
`xr = &y;`
`xr += 1;`
`cout << xr << " " << y;`	

`int x = 1;  int y = 3;`
`int *xr = &x;`
`xr = &y;`
`(*xr) += 1;`
`cout << *xr << " " << y;`	

`string s("Spoon");`
`string *sp = &s;`
`cout << sp.size();`	

4. Chasing Iterators

Consider the following for-loop that squares all of the even numbers within a vector:

```
vector<int> v = {1, 2, 3, 4, 5, 6};
for( size_t i = 0; i < v.size(); ++i ) {
    if ((v[i] % 2) == 0) {
        v[i] = v[i]*v[i];
    }
}
```

Rewrite the above loop but using iterators (ie using begin, end, vector<int>::iterator, etc.).

```
// YOUR CODE HERE
```

5. Best Function Names NA

Consider the following function definitions/declarations. If there are any issues, explain why. Assume all relevant headers are included, and we are using the standard namespace.

`double foo(int a, string s) {`
`  return s[a + 1];`
`}`
	`void bar(int x) {`
	`  return x + 1;`
	`}`

`char baz(string s) {`
`  return s[0];`
`}`
	`string garply(string str, int n) {`
	`  return s[n];`
	`}`

`void zzz() {`
`}`
	`string hmm() {`
	`  return 'a';`
	`}`

`double meow(int x) {`
`  return x;`
`}`
	`int bark(double d) {`
	`  return d;`
	`}`

6. Swpa-ing Numbesr

Part 1: Write a function swap1 that swaps the values of two int variables:

```
    int a = 42, b = 16;
    swap1(a, b);
    cout << a << " " << b; // Outputs: 16 42
// YOUR CODE HERE
```

Part 2: Next, consider the following function:

```
void swap2(int* x, int* y) {
    int tmp = *x;
    *x = *y;
    *y = tmp;
}
```

Say I have two integers a and b. How do I call swap2 to swap their values?

```
    int a = 2, b = 1;
    // YOUR CODE HERE



    cout << a << " " << b; // Outputs: 1 2
```

7. Not That Kind of Chunks...

Write a function `sort_chunks(vector<int>&v, int k)` that sorts each consecutive group of k elements, ie several "local" sorts, rather than a single global sort:

```
    vector<int> v = {4, 1, 2, 3, 0, 5, 9, 18, 1};
    // Sort each group of 3 elements
    sort_chunks(v, 3); // v is now: [1, 2, 4, 0, 3, 5, 1, 9, 18]

    vector<int> v2 = {4, 1, 2, 9, 1};
    sort_chunks(v2, 3); // v2 is now: [1, 2, 4, 1, 9]
```

Note: the length of v may not necessarily be a multiple of k - be sure to handle this case.
You should use the sort(left_iterator, right_iterator) function defined in <algorithm>:

```
    vector<int> v = {0, 3, 2, -1};
    sort(begin(v), end(v)); // v is now: [-1, 0, 2, 3]

void sort_chunks(vector<int>&v, int k) {
    // YOUR CODE HERE
```