Day 10

Multi-Dim arrays

Regrades

• If you want a quiz or homework regraded, you need to attach a sheet of paper to the front of the assignment that describes the error you believe exists.
  – We want them within a week of when they are returned. Regrades requested after that time will not be honored.

Multi-Dimensional arrays

• The idea is very simple.
  – We could have an array
    • int A[3][3]
    • That would be a 3x3 table of integers.
    • You can think of the row and column as being in either order.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-1</td>
<td>6</td>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

```c++
#include <iostream>
using namespace std;

const int SIZE = 3;

// A very useless program using multi-D arrays.
main()
{
  int B[SIZE][SIZE];
  int i=0;
  int j=0;
  
  B[2][2]=4;
  B[1][0]=6;
  B[0][1]=-5;
  B[0][0]=B[0][1]+B[2][2]*4;
  cout << B[0][0] << endl;
}
```
Issues with functions

• It turns out you need to specify the size of all but the first dimension in the parameter list.
  – You can specify them all if you wish
  – So:
    • void initArray(double A[][SIZE])
  – OR
    • void initArray(double A[SIZE][SIZE])

double time(double travel[][SIZE], int location[], int steps)
{
  int i=0;
  double tmp=0.0;
  int start, end;

  if(steps<2)
    return(0.0); // No place to go!
  while(i<(steps-1))
  {
    start=location[i];
    end=location[i+1];
    cout << i << " " << start << " " << end << endl;
    tmp=tmp+travel[start][end];
    i=i+1;
  }
  return(tmp);
}

main()
{
  int trip1[4]={0,2,3,0};
  int trip2[5]={0,1,2,1,0};
  double travel[SIZE][SIZE];

  initArray(travel);
  cout << "trip 1 takes: " << time(travel,trip1,4) << endl;
  cout << "trip 2 takes: " << time(travel,trip2,5) << endl;
}
**Scope**

- **General theme:**
  - Variables are only “visible” in the function (including main) in which they are declared.

- **Ramifications**
  - I can have two variables in different functions with the same name.
    - **They do not conflict.**
  - If you want to share information between functions, you need to pass it as an argument/parameter or as a return value.
  - In the debugger “out-of-scope” variables are not displayed.

**Scope**

- **Globals**
  - You can declare a variable to have global scope.
  - All functions (at least all in the same file) can use it.

- **Do not use globals**
  - One exception: global constants can be acceptable.

- To declare a global, just place it at the top of the file, outside of any function.

---

**Example from “wrong.cc” of P0**

```cpp
#include<iostream>
using namespace std;

const int STEPS=8;

double my_intpower(double value, int power)
{
    // more code goes here
}
```