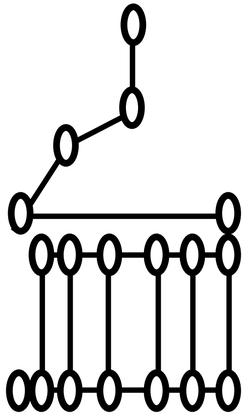
Lecture: Arrays, Vectors and Loops

ENGR 2730: Computers in Engineering



Five-Step Problem-Solving Methodology

1. State the problem clearly.
2. Describe the input and output.
3. Work hand examples.
4. Develop a solution/algorithm.
5. Test your solution.

Defining and initializing arrays

Initializing all elements to zero:

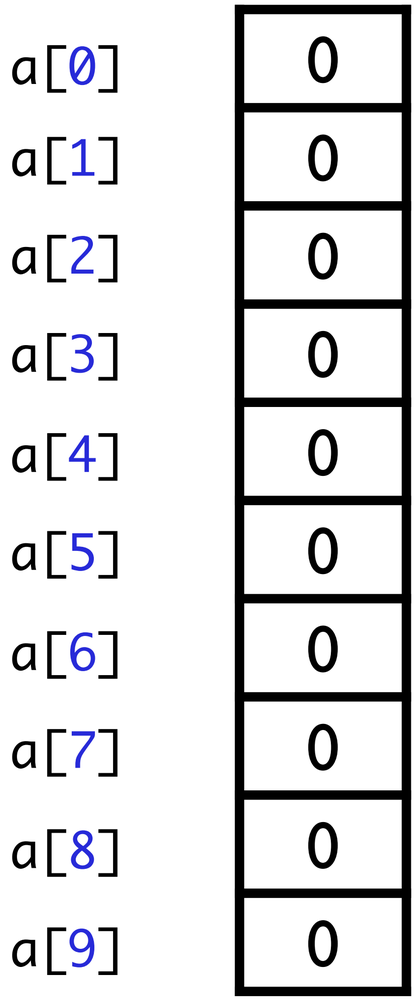
```
int a[10] = {0};
```

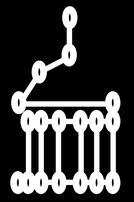
First element is set to zero.
All missing elements are set to zero.

Equivalently,

```
int a[10] = {};
```

All missing elements are set to zero.



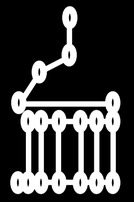


Write a program to compute the average of a list of numbers stored in an array.

```
#include <iostream>

using namespace std;

int main() {
    const int SIZE = 10;

    double data[SIZE] = {1.3, 2.7, 3.1, 4, 5, 6, 7, 8, -9.9, 10};

    double sum = 0;
    for(int i=0; i<SIZE; i++){
        sum += data[i];
    }

    cout << "ave = " << sum/SIZE << endl;
    return 0;
}
```

Note: SIZE is a constant.
Constants are all capital letters by convention.

Note: Use global constants in program. Global constants allow you to change a value in one place rather than multiple places.

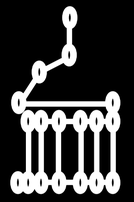


Write a program to compute the average of a list of numbers stored in an array.

```
#include <iostream>

using namespace std;

int main() {
    const int SIZE = 10;

    double data[SIZE] = {1.3, 2.7, 3.1, 4, 5, 6, 7, 8, -9.9, 10};

    double sum = 0;
    for(int i=0; i<SIZE; i++){
        sum += data[i];
    }

    cout << "ave = " << sum/SIZE << endl;
    return 0;
}
```

Note: Elements that are not initialized are set to zero.
Initializing too many elements is a compiler warning or error.

Note: array initializer syntax.



Write a program to compute the average of a list of numbers stored in an array.

```
#include <iostream>

using namespace std;

int main() {
    const int SIZE = 10;

    double data[SIZE] = {1.3, 2.7, 3.1, 4, 5, 6, 7, 8, -9.9, 10};

    double sum = 0;
    for(auto num:data){
        sum += num;
    }

    cout << "ave = " << sum/SIZE << endl;
    return 0;
}
```

New: C++ 11 range-based loop.
Note the lack of brackets in the for-loop.

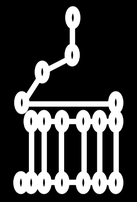
Vector Example

```
#include <vector>

int main() {
    vector<int> myNumbers; // create an empty vector of integers

    cout << "initial size of int vector = " << myNumbers.size() << endl; // prints: initial size of int vector = 0

    for (int n=0; n < 10; n++) {
        myNumbers.push_back(n); // add numbers to end of vector (vector automatically resized)
    }

    // print vector: option 1
    for (size_t i=0; i < myNumbers.size(); i++) {
        cout << myNumbers[i] << " ";    // prints: 0 1 2 3 4 5 6 7 8 9
    }
    cout << endl;

    // print vector: option 2
    for (size_t i=0; i < myNumbers.size(); i++) {
        cout << myNumbers.at(i) << " ";    // prints: 0 1 2 3 4 5 6 7 8 9
    }
    cout << endl;

    // modify elements of vector and print result
    for (size_t i=0; i < myNumbers.size(); i+=2) {
        myNumbers[i] = 0;     // sets even numbered elements to zero
    }

    for (auto item : myNumbers) {
        cout << item << " ";    // prints: 0 1 0 3 0 5 0 7 0 9
    }
    cout << endl;
```

(continued...)

Vector Example (con't)

```
    // remove last element and prints: size of integer vector after calling pop_back = 9
    myNumbers.pop_back();
    cout << "size of integer vector after calling pop_back = " << myNumbers.size() << endl;

    for (size_t i=0; i < myNumbers.size(); i++) {
        cout << myNumbers[i] << " ";          // prints: 0 1 0 3 0 5 0 7 0
    }
    cout << endl;


    // create vector of strings, with initial size of 3
    vector< string > myStrings(3);
    cout << "initial size of string vector = " << myStrings.size() << endl; // prints: initial size of string vector = 3


    myStrings.at(0) = "zero";
    myStrings[1] = "one";
    myStrings.at(2) = "two";
    myStrings.push_back("three");

    for (size_t i = 0; i < myStrings.size(); i++)
    {
        cout  << i << ": " << myStrings[i] << " "; // prints: 0: zero 1: one 2: two 3: three
    }
    cout << endl;

    return 0;
}
```



CQ: What is the output of the following program?

```
int main()
{
    int hist[5] = {0};
    int num_bins = 5;
    int indices[10] = {0, 0, 2, 3, 3, 3, 3, 4, 4, 4};
    int num_indices = 10;

    for (int i=0; i < num_indices; ++i)
    {
        hist[ indices[i] ]++;
    }

    for (int i=0; i < num_bins; ++i)
    {
        cout << hist[i] << " ";
    }
    cout << endl;
}
```

A. 0 0 0 0 0

B. 1 1 1 1 1

C. 2 0 1 4 3

D. 0 0 2 3 3

E. 2 3 4 4 4

State Machine Example

Manual controller for traffic light (Textbook Section 4.12)

The program declares a new enumeration type named LightState.

The program then declares a new variable lightVal of that type.

The loop updates lightVal based on the user's input.

The program moves among particular situations ("states") depending on input.

```
int main() {
   enum LightState {LS_RED, LS_GREEN, LS_DONE};
   char userCmd = '-';

   cout << "User commands: n (next), q (quit)." << endl << endl;

   LightState lightVal = LS_RED;
   while (lightVal != LS_DONE) {

      if (lightVal == LS_GREEN) {
         cout << "Green light  ";
         cin >> userCmd;
         if (userCmd == 'n') { // Next
            lightVal = LS_RED;
         }
      } else if (lightVal == LS_RED) {
         cout << "Red light  ";
         cin >> userCmd;
         if (userCmd == 'n') { // Next
            lightVal = LS_GREEN;
         }
      }

      if (userCmd == 'q') { // Quit
         lightVal = LS_DONE;
      }
   }

   cout << "Quit program." << endl;
   return 0;
}
```

```
User commands: n (next), r (red), q (quit).

Red light  k
Red light  l
Red light  n
Green light  n
Red light  q
Quit program.
```

ProjectEuler.net, Problem 1. Multiples of 3 and 5

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.