CS 101: Computer Programming and Utilization

Shivaram Kalyanakrishnan
(Abhiram Ranade's slides, borrowed and edited)
Lecture 16

Today’s Lecture

- 2D Arrays
- Random number generation
- Simulation: executing a random process

Two dimensional arrays

Matrices or tables can be stored using “two dimensional arrays”

`double xyz[m][n];`

- Creates `m*n` variables: `xyz[i][j]`, for `0 ≤ i < m`, `0 ≤ j < n`.
- `i`th row of `xyz`:

`xyz[i][0], xyz[i][1], … xyz[i][n-1]`

- `j`th column of `xyz`:

`xyz[0][j], xyz[1][j], … xyz[m-1][j]`

- `m,n`: first, second dimension of array `xyz`.
- Variables stored in memory in “row major” order:
  - row 0, followed by row 1, …, row `m-1`.

Two dimensional arrays (contd)

- Initialization possible:

```
int pqr[2][3]={{1,5,7},
               {13,6,2}};
```

- Values picked up from the initialization list in row major order.

- Better versions of two dimensional arrays are discussed in Chapter 22.

Example

- Create a 10x10 matrix A and initialize it to identity, i.e. value 1 in A[i][i] for all i and 0 elsewhere

```
double A[10][10];

for(int i=0; i<10; i++)

  for(int j=0; j<10; j++)

    if(i == j) A[i][j] = 1;

    else A[i][j] = 0;
```

Passing 2 dimensional arrays to functions

- Would be nice to have functions for matrices, tables, collections of strings...
- Example:
  - a function which multiplies an m x n matrix by an n x p matrix for any m, n, p
- The natural idea would be to pass
  - The starting address of the array
  - The number of rows
  - The number of columns.
- Unfortunately, C++ requires to specify the number of columns as a fixed number at the time of writing the function!
- This is a C++ language limitation.
  - Matrix multiply function above not possible
- Later we will see a more modern way which does not have this limitation.

Example

```
void printCountries(char c[][20], int n){
// We need to specify constant number of
columns.
// The number of rows can be specified; it is n.
// This function simply prints all the strings
in c.
  for(int i=0; i<n; i++)
    cout << c[i] << endl;
}
int main(){
  char countries[3][20]= {“India”, “Nepal”,
“China”};
  printCountries(countries, 3);
}
```

Today’s Lecture

- 2D Arrays
- Random number generation
- Simulation: executing a random process

rand() and randuv()

- rand() returns an integer drawn uniformly at random from the set {0, 1, 2, …, RAND_MAX}.

- randuv(x, y) implemented in s++ returns a real number drawn uniformly at random from the interval [0, 1].

- int seed = 0; srand(seed); sets a seed for the random number generator. For every seed, the sequence of numbers generated is deterministic.

- In computers, random = “pseudorandom”.

Today’s Lecture

- 2D Arrays
- Random number generation
- Simulation: executing a random process