CS 101 Computer Programming and Utilization

Lecture 10

Passing Arrays as parameters
Side effects and non-pure functions

Feb 11, 2011

Prof. R K Joshi
Computer Science and Engineering
IIT Bombay
Email: rkj@cse.iitb.ac.in

Revision: Separate Compilation; Decision trees

- Functions for reuse of code
- Define once call any no of times
- Definition and use in separate files
- These can be separately compiled
- And then linked
- Promotes reuse of function definition
- Promotes decomposition of software in terms of functional modules
- Builds and versions
- Decision trees
  - Identify variables that govern the logic
  - Identify conditions which are main ingredients in the logic
  - Make a decision tree to cover the entire problem space
  - One problem can be covered by many decision trees

Parameters

- Formal parameters
  - Appear inside definitions
  - Are variables with a type specified for each
- Actual parameters
  - Appear inside calls
  - Are values, variables, expression
- Names of formal parameters can be different from the names used in actual parameters.
- Parameters are by default passed by value (also called passed by copy)

Formal and actual Parameters

```
int f (float, char);
int f (float x, char y) {.....}  // x,y are formal
int main () {

  P = f (m, c);     // m, c are actual
  Q = f (3, c);      // 3, c are actual
  R = f (m, 'Y');   // so are m, 'Y'
  L = f (3, 'N');   // so are 3, 'N'
}
```

Pass by value

```
int func (int x) {
  X = 10;
  return x*x;
}
int main () {
  Int y;
  Cout << func (y) << endl;
 Cout << y << endl;  // what value of y?
}
```

Array as parameters

- An example declaration:

  int func (int A[ ], int n);

- A usage:

  x = func (A, size);

- Using index in definition

  For (i=0; i<size; i++) .... A[i] ...

What if a body of a function makes a change to an element in an array that is passed in?

```
Int func (int A[ ], int size) {
   A[size-1] =  1551;
}
Int main () {
   ..
   Func (B, n);
  Cout << B[n-1] << endl;  // will it change?
}
```

What really is 'A' in int A[5]?

- It's of course a name of the array
- But in C++, we also know that it is actually the starting location of the array
- Try

```
cout << A << endl;
```

- Int A[10] indeed means 10 integers located starting from location A.

Array variable passed by copy

```
int func (int A[], int size) {
 int B[size];
 … populate B ....
 A = B;  // will this have effect outside?
}
```