Computer Fundamentals

Dr. Safdar Nawaz Khan Marwat
DCSE, UET Peshawar

Lecture 19

safdar@uetpeshawar.edu.pk

Planning a Computer Program

- Plans
  - The steps to solve a problem
  - Describe expected results
  - Programming without a plan is difficult
  - Three planning tools discussed
    - Pseudocode
    - IPO chart
    - Flowchart

Planning Tools

- Pseudocode
  - Natural language statements that resemble code
  - Describes what must be done
  - Can be written by non programmers
  - Programmers develop unique versions

Planning Tools (cont.)

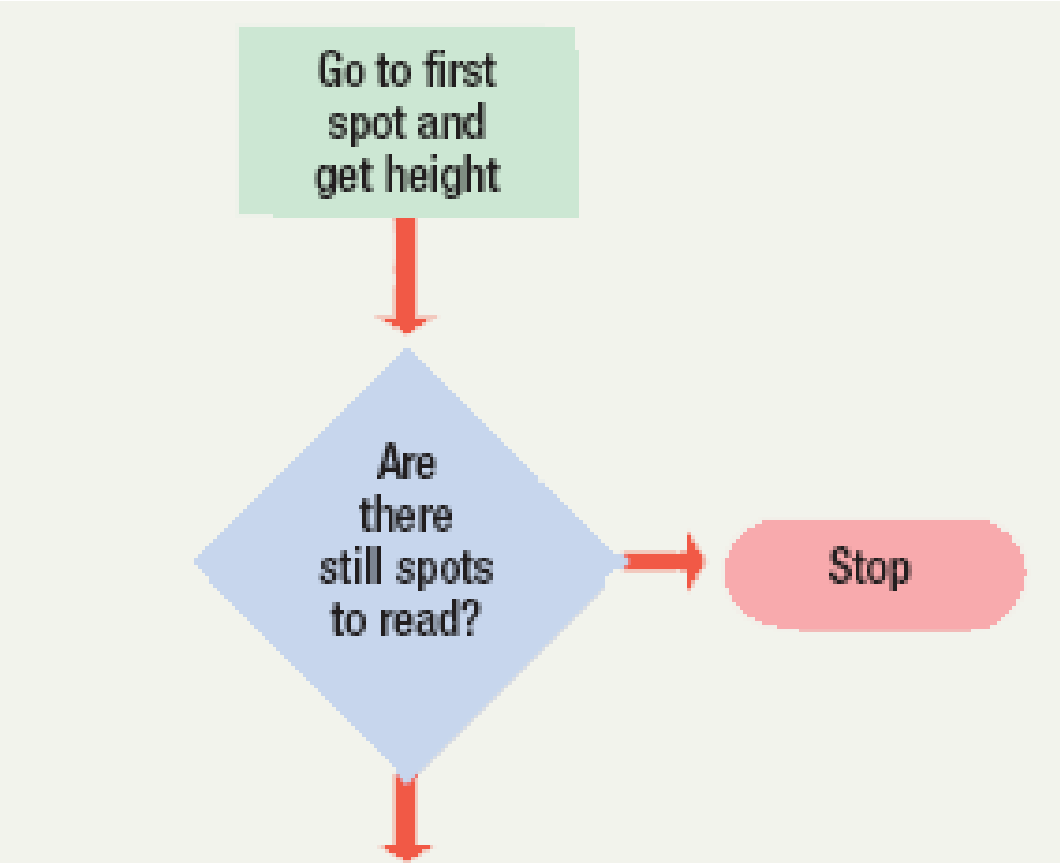
- Input-processing-output (IPO) charts
  - Determines what is needed
  - Input column
    - Data inputted by user
  - Processing column
    - Pseudocode describing problem solution
  - Output column
    - Desired output from program

The IPO Chart for a Program That Calculates Gross Pay for an Hourly Employee

Input	Processing	Output
Hours worked	Input hours worked	Gross pay
Hourly wage	Input hourly wage	
	Validate data	
	Pay = hours worked * hourly wage	
	Display gross pay	

Planning Tools (cont.)

- Flow Chart
  - Diagrammatic representation of program



How Programs Solve Problems (cont.)

- Algorithm
  - Set of steps to accomplish a task
    - Always leads to a solution
  - Steps are always the same
  - Flowcharts can describe algorithms
    - Structured tool for drawing algorithms
  - Algorithms appear in all programs

How Programs Solve Problems (cont.)

- Heuristic
  - Set of steps
    - Solution is usually found
  - Rule or method that helps solve problems faster than doing all computing
  - Solution may not be optimal
  - Used when algorithms fail
    - Algorithm is nonexistent or too complex
  - Appear in more complex applications
    - Data mining
      - Practice of examining large pre-existing databases in order to generate new information
    - Anti-virus software
      - File trying to write on hard disk, access emails

Structured Programming

- Programming using defined structures
- Creates easy to read code
- Programs are efficient and run fast
- Several defined structures

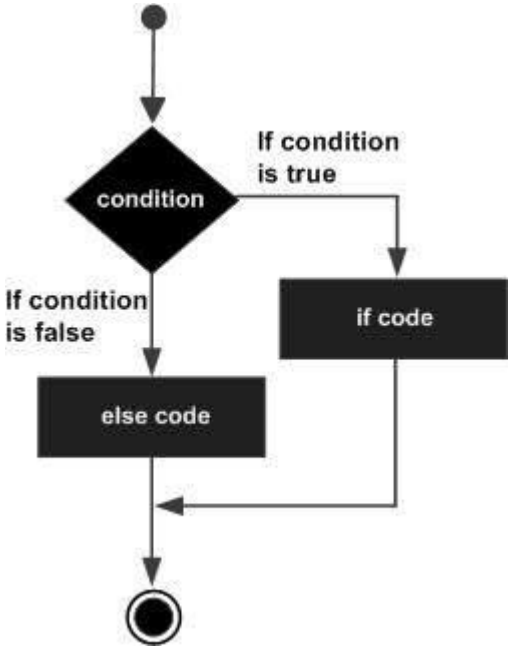
Structured Programming (cont.)

- Program control flow
  - Order in which program statements are executed
  - Typically executed in proper order
  - Branching statements allow multiple flows
  - Constructs can change the flow
    - Decision statements
    - Loops

Structured Programming (cont.)

- Selection statement
  - Also called conditional statement
  - Performs a true or false test
  - Determines which code to execute next

```
if(true)
        { // statement(s) will execute if the boolean expression is true }
else
        { // statement(s) will execute if the boolean expression is false }
```



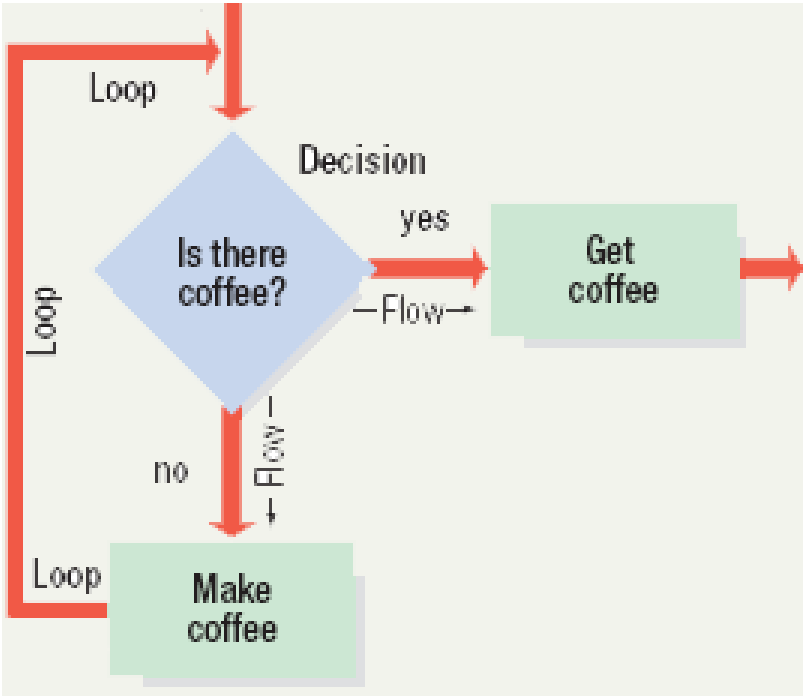
Structured Programming (cont.)

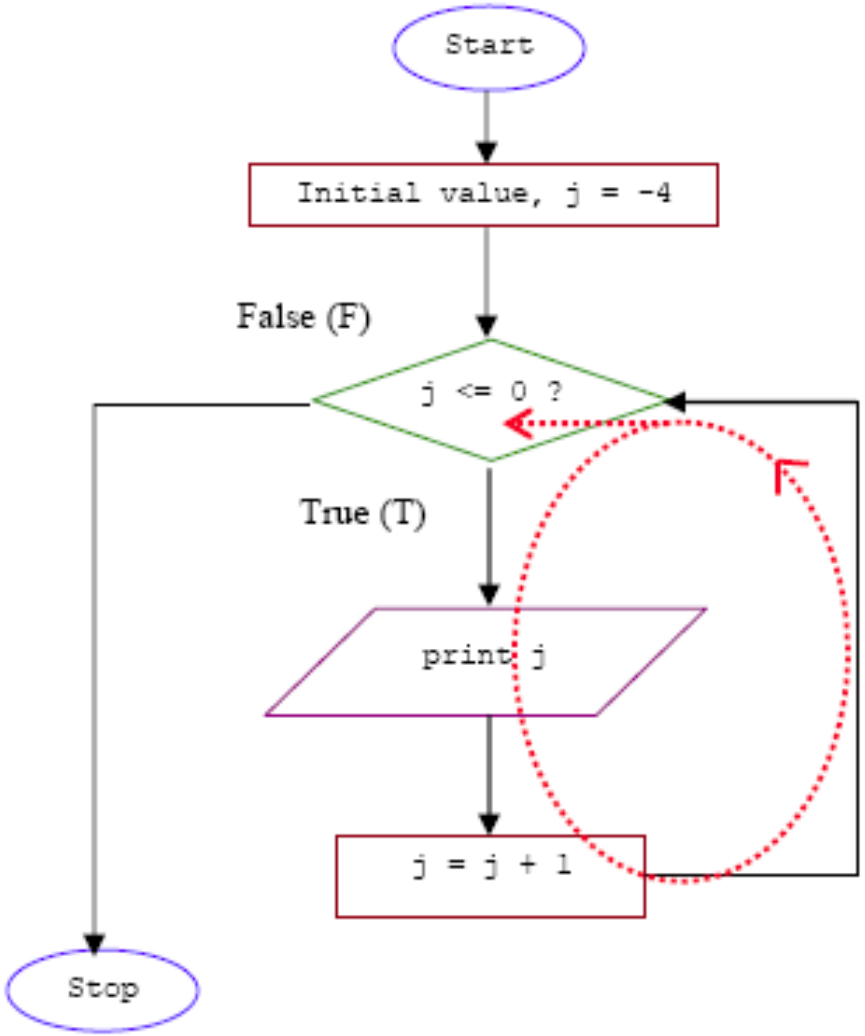
```
#include <iostream>
using namespace std;

int main ()
   { // local variable declaration:
   int a = 100; // check the boolean condition
   if( a < 20 )
      { // if condition is true then print the following
      cout << "a is less than 20" << endl;
      }
   else
      { // if condition is false then print the following
      cout << "a is not less than 20" << endl;
      }
   cout << "value of a is " << a << endl;
   return 0;
   }
```

Structured Programming (cont.)

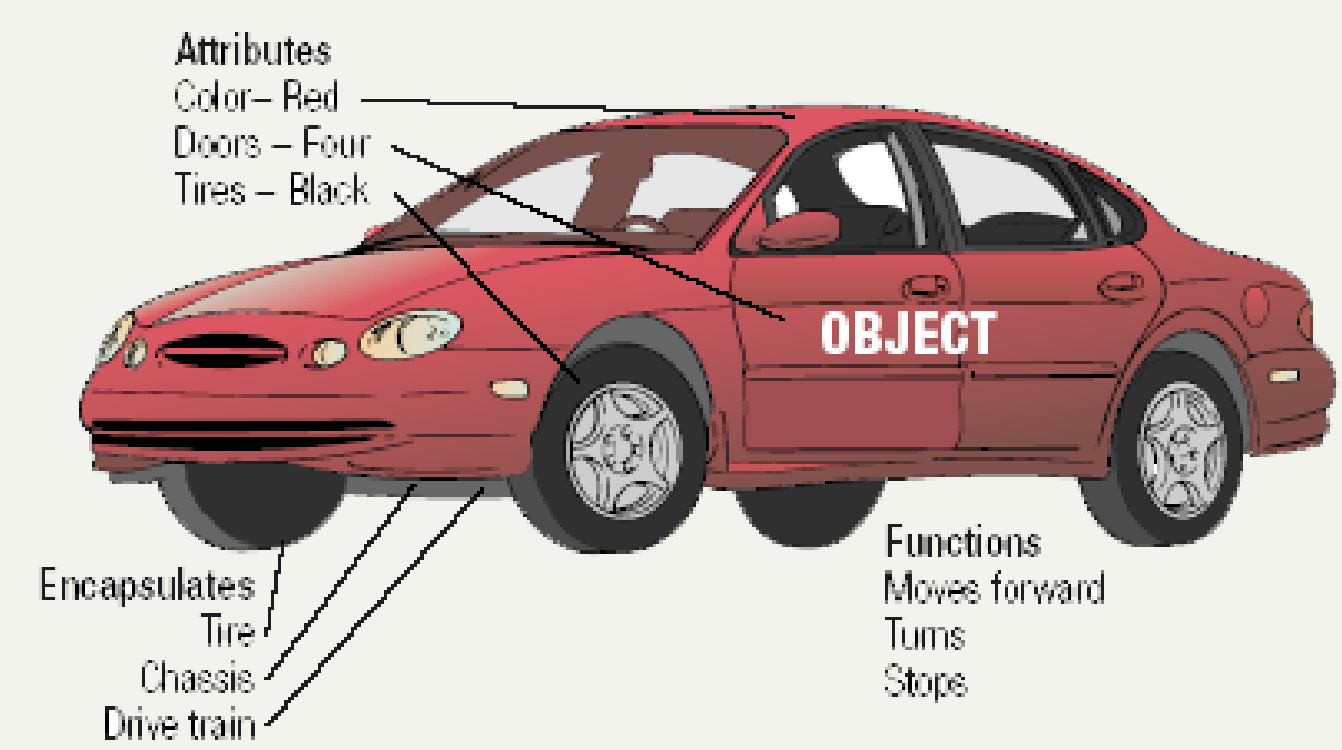
- Repetition statements
  - Also called looping structures
  - Repeats a section of code
    - Until an exit condition is reached





Object Oriented Programming

- Also known as OOP
- Enhances structured programming
- Intuitive method of programming



Object Oriented Programming (cont.)

- Class
  - Used to specify form of an object
  - Combines data representation and methods for manipulating that data into one neat package
  - Data and functions within a class called members of the class
- Object
  - Class provides the blueprints for objects
  - So object is created from class
  - Objects of a class declared with exactly same sort of declaration as that of basic variables

Object Oriented Programming (cont.)

- OOP develops objects
  - All real world items are objects
  - OOP develops code versions
    - Contains data about the item
    - Contains functionality
  - Object brings both into one package

Object Oriented Programming (cont.)

- Inheritance
- Code reuse
  - Code used in many projects
  - Speeds up program development
  - Simplifies program development

Object Oriented Programming (cont.)

- Encapsulation
  - Object Oriented Programming concept
  - Binds together data and functions that manipulate the data
  - Keeps both safe from outside interference and misuse
  - Data encapsulation led to important OOP concept of data hiding

Object Oriented Programming (cont.)

```
class Computer // Standard way of defining the class
  {
  public:
  // This means that all of the functions below this(and any variables)
  // are accessible to the rest of the program.
  // NOTE: That is a colon, NOT a semicolon...
  Computer(); // Constructor
  ~Computer(); // Destructor
  void setspeed ( int p );
  int readspeed();
  protected:
  // This means that all the variables under this, until a new type of
  // restriction is placed, will only  be accessible to other functions in the
  // class. NOTE: That is a colon, NOT a semicolon...
  int processorspeed;
  };
  // Do Not forget the trailing semi-colon
```