

## Software Tools

### Programming and Languages

## Programs

- Program - A set of step by step instructions that direct the computer to do a task that you want it to do and produce the results you want.
- Programming Language - A set of rules that instructs the computer what operations to perform.



R.M. Laurie - 2

## Programmers

- A programmer's job is to convert problem solutions into instructions for the computer.
- If the project is large, they may also need to coordinate the needs of users, managers, and systems managers.



R.M. Laurie - 3

## The Programming Process

- The steps involved in developing a program include:
  - Define the problem
  - Plan the solution
  - Code the program
  - Test the program
  - Document the program

R.M. Laurie - 4

## Define the Problem



- The programmer meets with a users to analyze the problem.
- Items like input, processing and output are discussed.

- Example: Convert time in 24 hour format to 12 hour time.
- 1432 → 2:32 pm

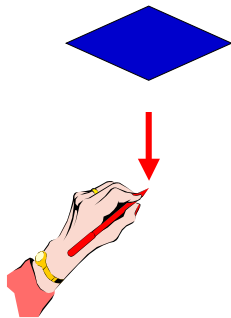
R.M. Laurie - 5

## Plan the Solution - Algorithm

- Flow Charts and Pseudocode are used to plan the solution.
- Flowchart - A pictorial representation of the ordered step by step process to solve the problem. Models instruction sequencing.
- Psuedocode - An English like language that can state your solution with more precision the just English but less precision than a programming language.

R.M. Laurie - 6

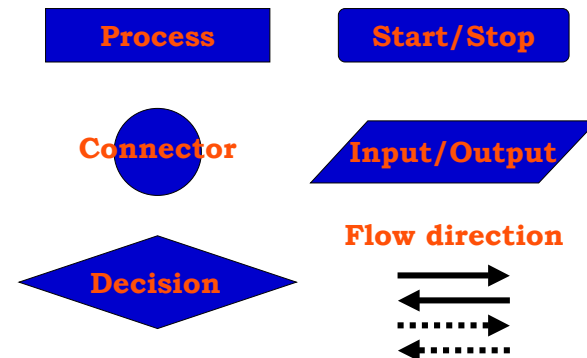
## Flowchart Basics



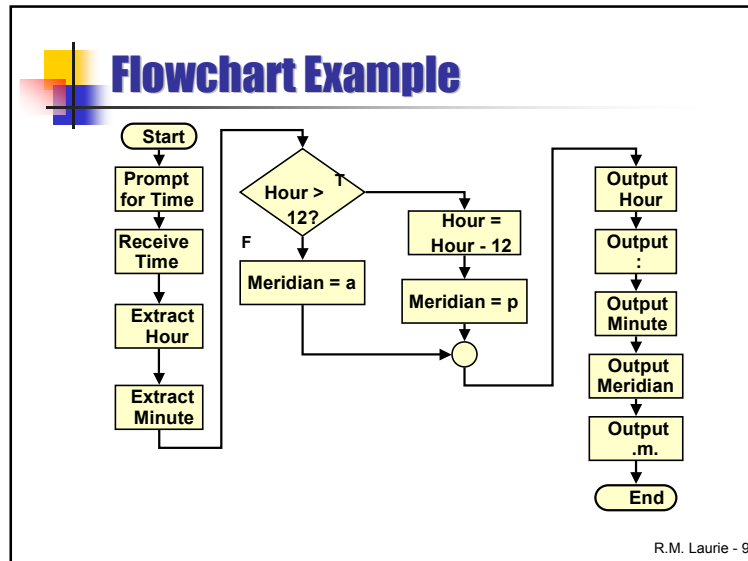
- A flowchart consists of arrows to represent direction the program takes and boxes and symbols to represent actions.

R.M. Laurie - 7

## Flowchart Symbols



R.M. Laurie - 8



### Desk-checking

- This form of testing involves mentally checking the logic of the program to ensure that it is error-free and workable using several sets of test data items.
- 345
- 1452
- 2300

R.M. Laurie - 10

### Code the Program

- Coding a program means to translate the logic from a flowchart or pseudocode into a programming language.
- The rules of computer languages are called **syntax**. These rules must be followed precisely.
- Common Languages are **BASIC, COBOL, PASCAL, FORTRAN, Java, and C++**.
- The program must be keyed in using a text editor (As with HTML).

R.M. Laurie - 11

### Example: C++ Program

```

#include <iostream>
using namespace std;
int main(void)
{
    int nEntry = 1, nHour, nMinute;
    char cAM = 'a';
    cout << "Enter the the 2400 hour time \n>";
    cin >> nEntry;
    nMinute = nEntry % 100;
    nHour = nEntry / 100;
    if(nHour > 12)
    {
        nHour = nHour - 12;
        cAM = 'p';
    }
    cout << nHour << ':';
    if(nMinute < 10) cout << '0';
    cout << nMinute << ' ' << cAM << ".\n\n";
    return 0;
}
    
```

R.M. Laurie - 12

## Testing the Program

- **Compiler** - Process of converting the keyed in program or source module into something the computer can understand called an executable module. Program instructions that the compiler does not understand are called *Syntax errors*.
- **Debugging** - Term used when detecting, locating, and correcting **BUGS**. When a program displays incorrect results for known test data these are called *Logic Errors*.

R.M. Laurie - 13

## Document the Program



- **Documentation** - A detailed written description of the programming cycle and specific facts about the program.
- This is an ongoing and necessary process.
- Manuals are one example.

R.M. Laurie - 14

## Generations of Programming Languages

There are several generations of programming languages:

1. **Machine**
2. **Assembly**
3. **High-level**
4. **Very high-level**
5. **Natural**

R.M. Laurie - 15

## First Generation: Machine Language

- This is the lowest level of programming language because it represents data and program instructions as 0s and 1s.
- All programming languages are eventually converted into machine language.

...		
Program Start	D000	86
	D001	12
	D002	8B
	D003	0C
	D004	B7
	D005	D1
	D006	00
	D007	BB
	D008	D1
	D009	10
	D00A	B7
	D00B	D1
	D00C	01
	...	
	FFFF	

R.M. Laurie - 16

## Second Generation: Assembly Language

- Assembly languages replace 0s and 1s with mnemonic codes.
- Each assembly instruction corresponds to one machine code instruction
- Requires an assembler to convert assembly source code to machine code

R.M. Laurie - 17

## Assembly Language Example

Address	Opcode	Operand	Assembly Code
D000	86	12	LDA #\$12
D002	8B	0C	ADDA #\$0C
D004	B7	D100	STA \$D100
D007	BB	D110	ADDA \$D110
D00A	B7	D101	STA \$D101
D00D	8B	1E	ADDA #\$1E
D00F	B7	D102	STA \$D102
D012	24	07	BCC \$D01B
D014	86	00	LDA #\$00
D016	B7	D110	STA \$D110
D019	20	0C	BRA \$D007
D01B	3F		SWI
D01C	08		FCB 08
Data Section			
D110	C0		

R.M. Laurie - 18

## Third Generation: High-Level Language

- High-level languages use English-like words that are much easier for humans to understand.
- A compiler is needed to convert the high-level language into machine language that computers understand.
- Examples: BASIC, COBOL, PASCAL, FORTRAN, Java, and C++.

R.M. Laurie - 19

## Third Generation: C++ Example

```
#include <iostream>
using namespace std;
int main(void)
{
    int nEntry = 1, nHour, nMinute;
    char cAM = 'a';
    cout << "Enter the the 2400 hour time \n>";
    cin >> nEntry;
    nMinute = nEntry % 100;
    nHour = nEntry / 100;
    if(nHour > 12)
    {
        nHour = nHour - 12;
        cAM = 'p';
    }
    cout << nHour << ':';
    if(nMinute < 10) cout << '0';
    cout << nMinute << ' ' << cAM << ".m.\n\n";
    return 0;
}
```

R.M. Laurie - 20

### Fourth Generation: Very High-Level Language

- Fourth generation languages are often known as 4GLs.
- 4GLs are a shorthand programming language that is about 10 times more productive than third generation languages.
- Examples: SQL, Robot Control Languages

R.M. Laurie - 21

### Fourth Generation: SQL Example

```
TABLE FILE SALES
SUM UNITS BY MONTH
    BY CUSTOMER BY PRODUCT
ON CUSTOMER SUBTOTAL
PAGE BREAK
END
```

R.M. Laurie - 22

### Fifth Generation: Natural Language



- This generation of programming languages more resembles “natural” spoken English.
- The user communicates with the computer by speaking.
- Example: Dragon Natural Speaking Software

R.M. Laurie - 23

### Major Programming Languages

There are several languages with which to write your program:

- FORTRAN – FORMula TRANslator language primarily used by engineers and scientists.
- COBOL – COBOL - COmmon Business-Oriented Language. Most commonly used language in business.
- BASIC – Beginners All-purpose Symbolic Language. Widely used by beginners since it easy to learn.
- Pascal – Used in colleges as a training language.
- ADA - Military languages used for weapons systems
- C++ – Most Common General Purpose (OOP)
- Java – Internet Programming Language (OOP)

R.M. Laurie - 24



## Fortran

```
1. ...
2. WRITE (6,10)
3.   SUM=0
4.   COUNTER = 0
5.   WRITE (6,60)
6.   READ (5,40) NUMBER
7. 1 IF (NUMBER .EQ. 999) GOTO 2
8.   SUM = SUM + NUMBER
9.   GOTO 1
10. ...
```

R.M. Laurie - 25



## COBOL

```
1. ...
2. 300-INITIALIZATION-ROUTINE.
3.     DISPLAY "PLEASE ENTER A NUMBER".
4.     ACCEPT NUMBER-ITEM.
5. 400-ENTER-AND-ADD.
6.     ADD NUMBER-ITEM TO SUM-ITEM.
7.     ADD 1 TO COUNTER.
8.     DISPLAY "PLEASE ENTER NEXT NUMBER"
9. ...
```

R.M. Laurie - 26



## BASIC

```
1. ...
2.   SUM = 0
3.   COUNTER = 0
4.   PRINT "PLEASE ENTER A NUMBER"
5.   INPUT NUMBER
6.   DO WHILE NUMBER <> 999
7.     SUM = SUM + NUMBER
8.     COUNTER = COUNTER + 1
9. ...
```

R.M. Laurie - 27



## Pascal

```
1. ...
2. BEGIN
3.   sum :=0;
4.   counter :=0;
5.   WRITELN ('PLEASE ENTER A NUMBER');
6.   READLN (number);
7.   WHILE number <> 999 DO
8.     Begin (*while loop*)
9.       sum := sum + number;
10. ...
```

R.M. Laurie - 28

## C++

1. ...
2. `cout << "PLEASE ENTER A NUMBER";`
3. `cin >> number;`
4. `while (number != 999)`
5. `{`
6.     `sum := sum + number;`
7.     `counter ++;`
8.     `cout << "\nPlease enter the next number";`
9. `...}`

R.M. Laurie - 29

## Object-Oriented Programming

- Object oriented languages: C++, Java, Visual Basic, Delphi
- Different then procedural languages in that program code is not executed sequentially but is event driven.
- Usually used to write Windows type programs using the mouse buttons and keyboard click events.

R.M. Laurie - 30

## Classes and Objects

- Class specifies type of objects
  - Properties: Data, Facts, Characteristics, Attributes
  - Methods: Functions, Instructions
- Object is an instance (occurrence) of a class
- Encapsulation
  - Describes Object contains both properties and methods.
  - Everything it needs to exist as an object

R.M. Laurie - 31

## More OOP Terminology

- Sub-Class: Class with more specific properties. Also includes class properties
- Inheritance: Sub-class possesses (inherits) all properties of derived class

R.M. Laurie - 32