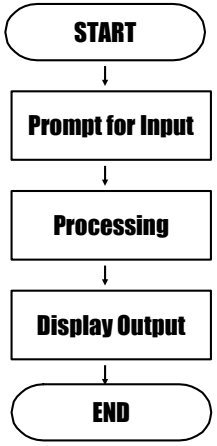
Control Structures

- Flow of control
  - Definition: The execution sequence of C++ program statements
- Sequential Control Structure
  - What we have been doing Start to End
- Selection (Branching) Control Structure
  - Conditional T/F decision processing
  - Relational and Logical Operators
- Repetition (Loop) Control Structure
  - Repetition processing

 1

Sequential Control Structure



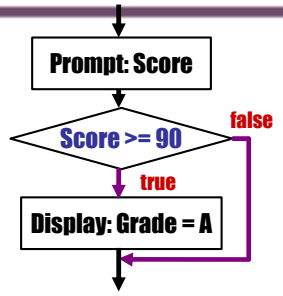
- Simply the execution of program instructions from the top to bottom
- Characterized by a flow chart construct without branches

```
START
  Prompt: Input
  Processing
  Display: Output
END
```

 2

Selection Control Structure

- Selects one of two possible paths for program statements
- Utilized for making decisions based on evaluating a true/false Assertion
- Flowcharts use a diamond shaped construct, containing a (true or false) *assertion* with two possible *branches*



Pseudo code:

```
START
   Prompt: Score
   IF Score >= 90 THEN
      Display: Grade=A
   ENDIF
END
```

Assertion Expression

- An Assertion is a conditional expression or question that is either true or false
  - Question evaluated *True* or *False* (Humans)
  - Question as *Yes* or *No* (Humans)
  - Evaluated as 1 or 0 (Computers)
- Examples:
  - Is Score greater then or equal to 90?
  - Is Entry == 'Y'?
  - Do you like Chocolate?
  - Is Guess == Random Number?
  - ~~What is your Score on the test?~~
  - ~~How much do you like Chocolate?~~

 4

Relational Operators

- Relational operators are used to compare two *data objects*
- The result of the comparison is either true (1) or false (0)
- There are six Relational Operators:

==	Equal To	!=	Not Equal To
>	Greater	>=	Greater or Equal
<	Less	<=	Less or Equal

- Note the difference:

==	Equal To
=	Assignment

 5

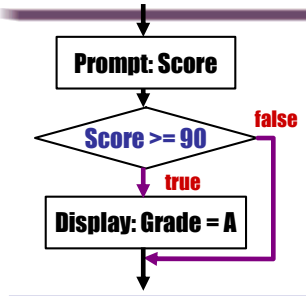
More Operators and Precedence

(Highest to Lowest)

( )	Defines order of operation
! ++ -- -	Logical NOT, Unary Arithmetic
sizeof()	How many bytes?
* / %	Multiplication, Division, Modulus
+ -	Addition, Subtraction
< <= > >=	Relational Operators
== !=	
&&	Logical AND
||	Logical OR
=	Assignment
,	Comma Operator

 6

if( ) Control Structure



Design Phase

Pseudo code:

```
START
   Prompt: Score
   IF Score >= 90 THEN
      Display: Grade=A
   ENDIF
END
```

Implementation Phase

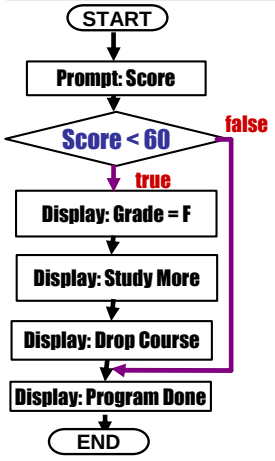
C++ code:

```
/**********************************
* PROGRAM: GetA.cpp
**********************************
/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Did you get an A\?\n\n";

  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore >= 90)
    cout << "You got an A\n";
  // system("pause");
  return 0;
}
```

 7

Compound if( ) Control Structure




```
/**********************************
* PROGRAM: ifCompoundF.cpp
**********************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore < 60)
  {
    cout << "You have an F.\n";
    cout << "Study More!\n";
    cout << "Drop the Course.\n\n";
  }
  cout << "Done\n\n";
  // system("pause");
  return 0;
}
```

Compound if( ) Control Structure

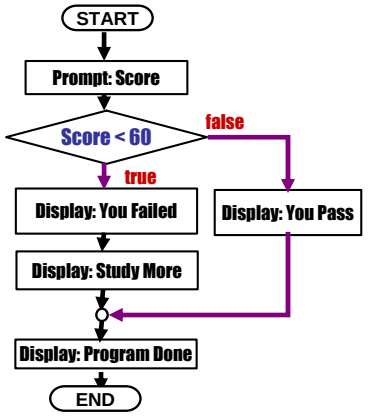
Pseudo code:

```
START
  Prompt: Score
  IF Score < 60 THEN
    Display: Grade=F
    Display: Study More
    Display: Drop Course
  ENDIF
  Display: Done
END
```

```
/*********************************
* PROGRAM: ifCompoundF.cpp
*********************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore < 60)
  {
    cout << "You have an F.\n";
    cout << "Study More!\n";
    cout << "Drop the Course.\n\n";
  }
  cout << "Done\n\n";
    // system("pause");
  return 0;
}
```

if( ) - else Control Structure

❖Two different execution path sequences

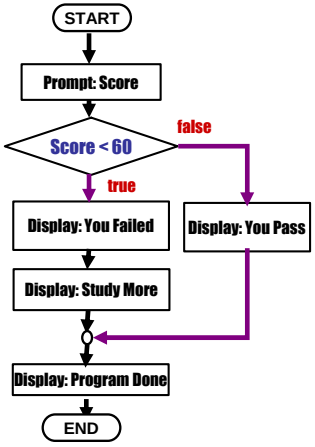


Pseudo code:

```
START
   Prompt: Score
   IF Score < 60 THEN
      Display: Grade =F
      Display: Study More
   ELSE
      Display: You Pass
   ENDIF
   Display: Done
END
```

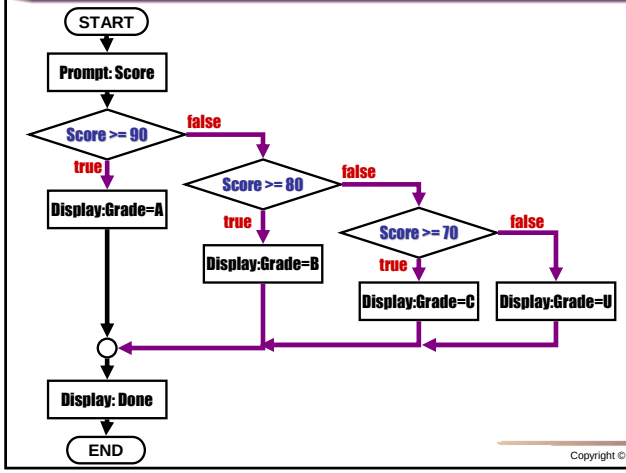
 10

if( ) - else Control Structure



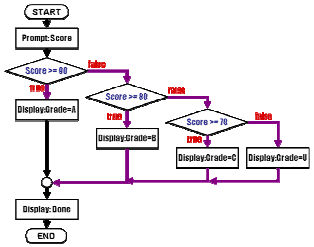

```
/*********************************
* PROGRAM: IfElse.cpp
*********************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore < 60)
  {
    cout << "You Failed.\n";
    cout << "Study More.\n\n";
  }
  else
    cout << "You Passed.\n\n";
  cout << "Done\n\n";
  // system("pause");
  return 0;
}
```

Nested if( ) – else Control Structure



 12

Nested if( ) – else C

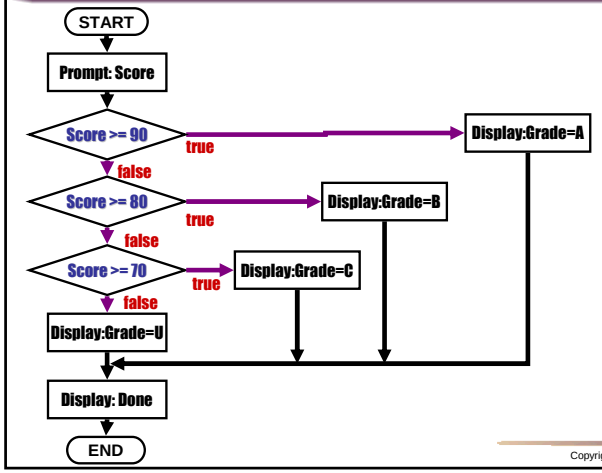



```
/*********************************
* PROGRAM: Grade.cpp
*********************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore >= 90)
    cout << "Grade=A";
  else
    if(nScore >= 80)
      cout << "Grade=B";
    else
      if(nScore >= 70)
        cout << "Grade=C";
      else
        cout << "Grade=U";
  cout << "\n\nDone\n\n";
  // system("pause");
  return 0;
}
```

```
/*************************
* PROGRAM: Grade.cpp
*************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore >= 90)
    cout << "Grade=A";
  else
    if(nScore >= 80)
      cout << "Grade=B";
    else
      if(nScore >= 70)
        cout << "Grade=C";
      else
        cout << "Grade=U";
  cout << "\n\nDone\n\n";
  // system("pause");
  return 0;
}
```

```
/*************************
* PROGRAM: Grade.cpp
*************************/
#include <iostream>
using namespace std;
int main()
{
  int nScore;
  cout << "Enter score\n>";
  cin >> nScore;
  if(nScore >= 90)
    cout << "Grade=A";
  else if(nScore >= 80)
    cout << "Grade=B";
  else if(nScore >= 70)
    cout << "Grade=C";
  else  cout << "Grade=U";
  cout << "\n\nDone\n\n";
  // system("pause");
  return 0;
}
```

if( ) – else if( ) – else Control Structure



 15

Creating Fault Tolerant Input

- `cin >> cChar;`
  - Reads first non-whitespace character
  - Extra characters left on input stream
- `cin >> nRadius;`
  - Reads first numeric character until last
  - Extra characters left on input stream
- `cin.ignore(100,'\n');`
  - Flushes next 100 characters or until newline character is reached from input stream
- A better solution for character input

```
cin >> cQuestion;
cin.ignore(100,'\n');
```

- A better solution for numerical input

```
cin >> nTemperature;
cin.ignore(100,'\n');
```

 16