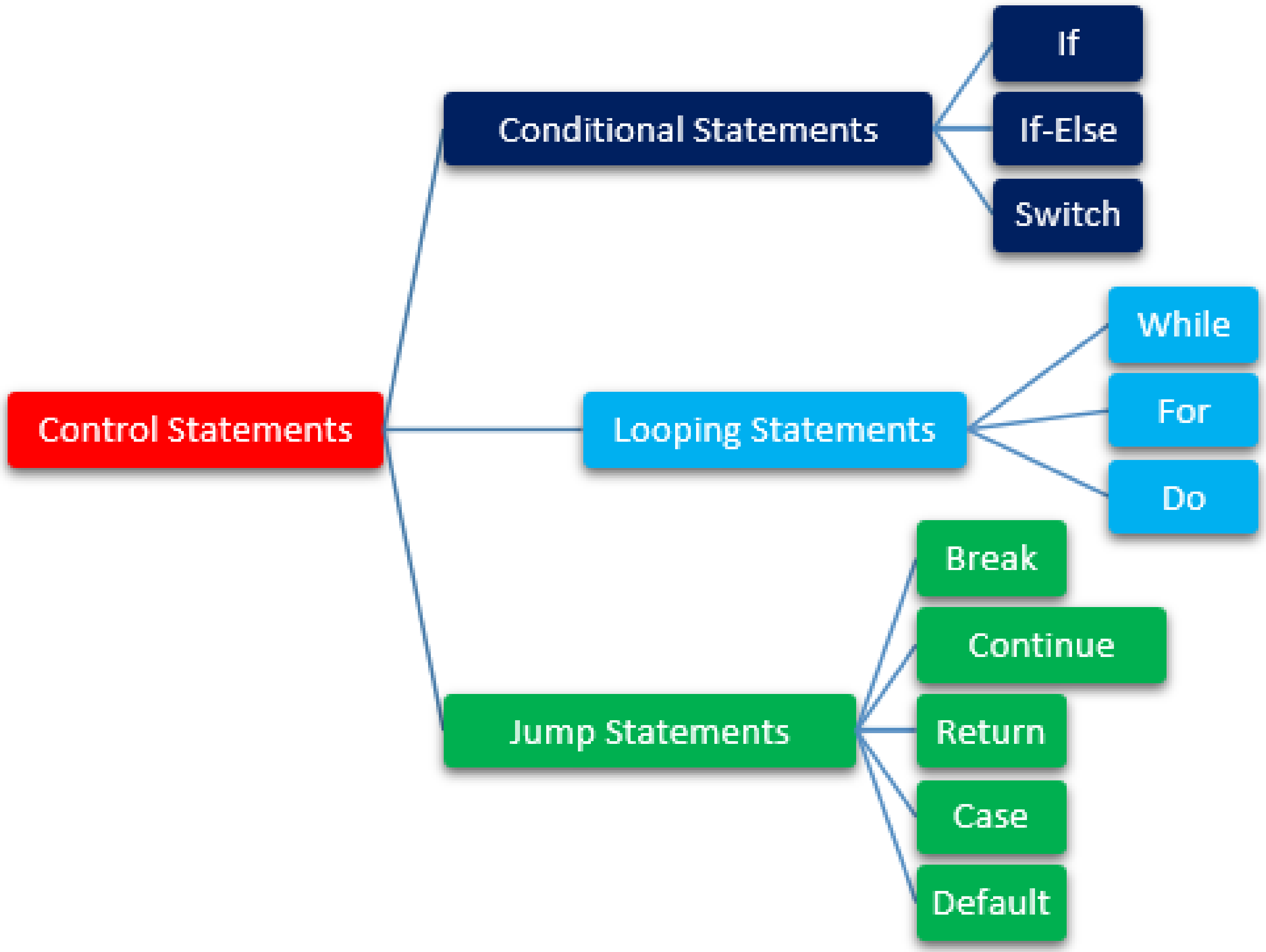
Loops and Conditional Statements

Control flow and branching using keywords,

such as **if**, **for**, and **while**

Loops and Conditional Statements



Conditional statements with the proper **comparison** and **boolean operators** allow the creation of **alternate execution paths** in the code.

Loops allow **repeated execution** of the **same** set of **statements** on all the objects within a sequence.

Jump statements allow you to **exit a loop**, start the next iteration of a loop, or explicitly **transfer program control** to a specified location in your program.

FOR Loop

for loop to repeat specified number of times

```
for index = values
   statements
end
```

Decrement Values

```
for v = 1.0:-0.2:0.0
   disp(v)
end
```

Execute Statements for Specified Values

```
for v = [1 5 8 17]
   disp(v)
end
```

Repeat Statements for Each Matrix Column

```
for I = eye(4,3)
    disp('Current unit vector:')
    disp(I)
end
```

Assign Matrix Values

```
s = 10;
H = zeros(s);

for c = 1:s
    for r = 1:s
        H(r,c) = 1/(r+c-1);
    end
end
```

Selectively Display Values in Loop : continue

```
for n = 1:50
    if mod(n,7)
        continue
    end
    disp(['Divisible by 7: ' num2str(n)])
end
```

WHILE Loop

while loop to repeat when condition is true

```
while expression
    statements
end
```

Repeat Statements Until Expression Is False

```
n = 10;
f = n;
while n > 1
    n = n-1;
    f = f*n;
end
disp(['n! = ' num2str(f)])
```

Exit Loop Before Expression Is False

```
limit = 0.8;
s = 0;

while 1
    tmp = rand;
    if tmp > limit
        break
    end
    s = s + tmp;
end
```

Conditional statements

Conditional statements enable you to select at run time which block of code to execute.

Some examples :

```
% Generate a random number
a = rand(100, 1);

% If it is even, divide by 2
if rem(a, 2) == 0
    disp('a is even')
    b = a/2;
end
```

```
a = rand(100, 1);

if a < 30
    disp('small')
elseif a < 80
    disp('medium')
else
    disp('large')
end
```

```
[dayNum, dayString] = weekday(date, 'long', 'en_US');

switch dayString
   case 'Monday'
      disp('Start of the work week')
   case 'Tuesday'
      disp('Day 2')
   case 'Wednesday'
      disp('Day 3')
   case 'Thursday'
      disp('Day 4')
   case 'Friday'
      disp('Last day of the work week')
   otherwise
      disp('Weekend!')
end
```

return

return the control to the invoking program before it reaches the end of the script or function.

Return Control to Keyboard

```
function idx = findSqrRootIndex(target,arrayToSearch)
idx = NaN;
if target < 0
   return
end
for idx = 1:length(arrayToSearch)
    if arrayToSearch(idx) == sqrt(target)
        return
    end
end
```

```
>>A = [3 7 28 14 42 9 0];
>>b = 81;
>> findSqrRootIndex(b,A)
```

Return Control to Invoking Function

```
function returnControlExample(target)
    arrayToSearch = [3 7 28 14 42 9 0];
    idx = findSqrRootIndex(target,arrayToSearch);

    if isnan(idx)
        disp('Square root not found.')
    else
        disp(['Square root found at index ' num2str(idx)])
    end
end
```

```
>> returnControlExample(49)
>> Square root found at index 2
```