Library Functions

- Functions that can be called in program
- Requires inclusion of header file at beginning of program
  - #include <iostream>
  - #include <cmath>
  - #include <cstdlib>
  - #include <ctime>
  - #include <cstring>
  - using namespace std;
- Header file provides declaration or *"prototype"* of the function for including
- See BCB5.HLP Borland help file for details

 1

cmath Functions

- abs
- acos, acosl
- cosh, coshl
- sin, sinl
- tan, tanl
- pow, powl
- sqrt, sqrtl

Header File
cmath was math.h in old C++

Syntax
#include <math.h>
double pow(double x, double y);

Description
Calculates x to the power of y. arguments and returns a long double result.

Return Value
On success, pow return the value calculated of x to the power of y.

 2

sqrt( ) Function

#include <cmath>
double sqrt(double x);
long double sqrtl(long double x);

Description
Calculates the positive square root.
sqrt calculates the positive square root of argument x.
sqrtl is the long double version; it takes a long double argument and returns a long double result.

Return Value
On success, sqrt and sqrtl return the value calculated, the square root of x. If x is real and positive, the result is positive. If x is real and negative, the global variable *errno* is set to EDOM Domain error

 3

Library Function Example

```
#include <cmath>
#include <iostream>
using namespace std;
int main(void)
{
  double dA, dB=4.0;
  dA = sqrt(dB);
  cout << dA << endl;
  dA = sqrt(dA);
  cout << dA << endl;
  dA = pow(pow(dA, dB), 3);
  cout << dA << endl << sqrt(dA) end;
  return 0;
}
```

← Declares Library Functions

```
2
1.41421
64
8
```

 4

cstdlib Functions

- srand((unsigned) time(NULL));
- int rand(void);

int rand(void);
Description
Random number generator. Uses a multiplicative congruential random number generator with period 2 to the 32nd power to return successive pseudo-random numbers.
Return Value
rand returns the generated pseudo-random number.

time_t time(NULL);
Description
time gives seconds, elapsed since 00:00:00 GMT, January 1, 1970.
Return Value
time returns the elapsed time in seconds.

void srand(unsigned seed);
Description
Initializes random number generator. It can be set to a new starting point by calling srand with a given seed number.
Return Value
None

 5

Random Number Example

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main()
{
    int nNum, nI;
    srand((unsigned) time(NULL));
    for(nI = 0; nI < 20; nI++)
    {
        nNum = rand();
        cout << nNum%10 << ", "
            << nNum << endl;
    }
    return 0;
}
```

```
4, 15334
2, 29412
8, 4148
1, 25881
9, 17629
8, 7018
4, 17064
7, 26347
0, 7200
7, 1717
7, 6087
0, 25550
8, 27888
9, 2329
9, 20579
2, 8162
0, 5950
0, 15820
7, 10667
```

```
7, 24117
7, 9097
6, 8576
0, 18870
3, 1783
3, 6393
5, 855
1, 10751
8, 28658
4, 28354
2, 17672
1, 15371
7, 21427
4, 6164
5, 25745
4, 16174
0, 2120
0, 5780
3, 4503
```

 6

Arrays

- Array
  - Grouping of similarly named variables
  - Grouped sequentially in memory
  - Accessed by using both their identifier and element number
    - 0 to one less then the total number of elements
- Dimension
  - The total number of elements of an array
  - Specified in the declaration
  - No *bounds checking* in C++

Counter[0]	30
Counter[1]	45
Counter[2]	53
Counter[3]	2
Counter[4]	879

 7

One Dimensional Arrays

- Declaration
  `int nArray[5];`
  - Reserves array memory nArray[0] to nArray[4]
- For an array dimension of n
  `int nArray[n];`
- Elements may be declared of any data type: int, float, char, double, long

nArray[0]	3423
nArray[1]	9441
nArray[2]	0016
nArray[3]	0348
nArray[4]	3400

 8

Initializing Arrays

- Initialization when declaring:

```
int nArray[5] = {0, 0, 0, 0, 0};
```

OR

```
int nArray[] = {0, 0, 0, 0, 0};
```

- Initialize array using for loop

```
for(nI=0; nI<5; nI++)
   nArray[nI] = 0;
```

- Elements are always numbered nArray[0] to nArray[n-1]

nArray[0]	0
nArray[1]	0
nArray[2]	0
nArray[3]	0
nArray[4]	0

 9

Array Bounds

- No *bounds checking* in C++
  - It is not a syntax error to assign a value to element nArray[5] or nArray[10]
  - However this memory location is not assigned to be part of the array
- Assigning value to *out-of-bounds* element has *Catastrophic* results
  - nArray[10] = 0;
  - Program may crash
  - Program may alter data

nArray[0]	0
nArray[1]	0
nArray[2]	0
nArray[3]	0
nArray[4]	0

 10

```
1.  #include <iostream>
2.  using namespace std;
3.  int main()
4.  {
5.      int nI, nScore, nGrdCnt[5];
6.      char cGrade[] = {'F', 'D', 'C', 'B', 'A'};
7.      for(nI = 0; nI < 5; nI++)
8.          nGrdCnt[nI] = 0;
9.      cout << "This program will calculate the grade \n"
10.         << "distribution for a series of entered scores.\n\n";
11.     while(true)
12.     {
13.         cout << "                (-1 to end)\rEnter score: ";
14.         cin >> nScore;
15.         if(nScore < 0)break;
16.         if(nScore >= 90)nGrdCnt[4]++;
17.         else if(nScore >= 80)nGrdCnt[3]++;
18.         else if(nScore >= 70)nGrdCnt[2]++;
19.         else if(nScore >= 60)nGrdCnt[1]++;
20.         else nGrdCnt[0]++;
21.     }
22.     cout << "\nRESULTS:\n----------------";
23.     for(nI=4; nI>=0; nI--)
24.         cout << endl << cGrade[nI] <<"\'s = " << nGrdCnt[nI];
25.     cout << "\n\nDone";
26.     return 0;
27. }
```

```
Enter score: 95
Enter score: 65
Enter score: -1

RESULTS:
----------------
A's = 1
B's = 0
C's = 0
D's = 1
F's = 0
```

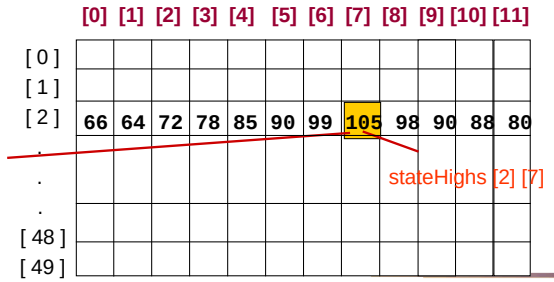
Multidimensional Arrays

- Two dimensional Arrays
  - float fArray[4][2];
  - rows, columns
  - Visualize like a Table
- Three dimensional Arrays
  - int nArray[6][4][2]
  - height, width, depth
  - Visualize like a 3D brick
- There is no limit to the dimensions of an Array in C++

 12

EXAMPLE – 2D Array to keep monthly high temperatures for all 50 states in one array.

```
const int NUM_STATES  = 50 ;
const int NUM_MONTHS  = 12 ;
int stateHighs [ NUM_STATES ] [ NUM_MONTHS ] ;
```



 13

```
Enter the X Y coordinates:1
   Point 1: X = 12
   Point 1: Y = 18
Enter the X Y coordinates:2
   Point 2: X = 18
   Point 2: Y = 26
_____________________________
Lenth of the line is 10
Point 1:  X=12  Y=18
Point 2:  X=18  Y=26
```

```
1.  #include <iostream>
2.  #include <cmath>
3.  using namespace std;
4.  int main()
5.  {
6.      int nI;
7.      double dLength, dDeltaX, dDeltaY, dLine[2][2] ;
8.      // Point 1 = [X1 Y1]
9.      // Point 2 = [X2 Y2]
10.     for(nI=0; nI < 2; nI++)
11.     {
12.         cout << "Enter the X Y coordinates:" << nI+1;
13.         cout << "\n   Point " << nI+1 << ": X = ";
14.         cin >> dLine[nI][0];
15.         cout << "   Point " << nI+1 << ": Y = ";
16.         cin >> dLine[nI][1];
17.     }
18.     dDeltaX = dLine[0][0] - dLine[1][0];
19.     dDeltaY = dLine[0][1] - dLine[1][1];
20.     dLength = sqrt(pow(dDeltaX, 2) + pow(dDeltaY, 2));
21.     cout << "_____________________________________";
22.     cout << "\nLenth of the line is " << dLength;
23.     cout << "\nPoint 1:  X=" << dLine[0][0] << "  Y=" << dLine[0][1];
24.     cout << "\nPoint 2:  X=" << dLine[1][0] << "  Y=" << dLine[1][1];
25.     return 0;
26. }
```