Introduction to Computing

Lecture 11

Dr. Naveed Anwar Bhatti

Webpage: naveedanwarbhatti.github.io

2D-Arrays

Two-Dimensional Arrays

Definition:

- A two-dimensional array is a list of one-dimensional arrays
- The general form of a two-dimensional array declaration is:

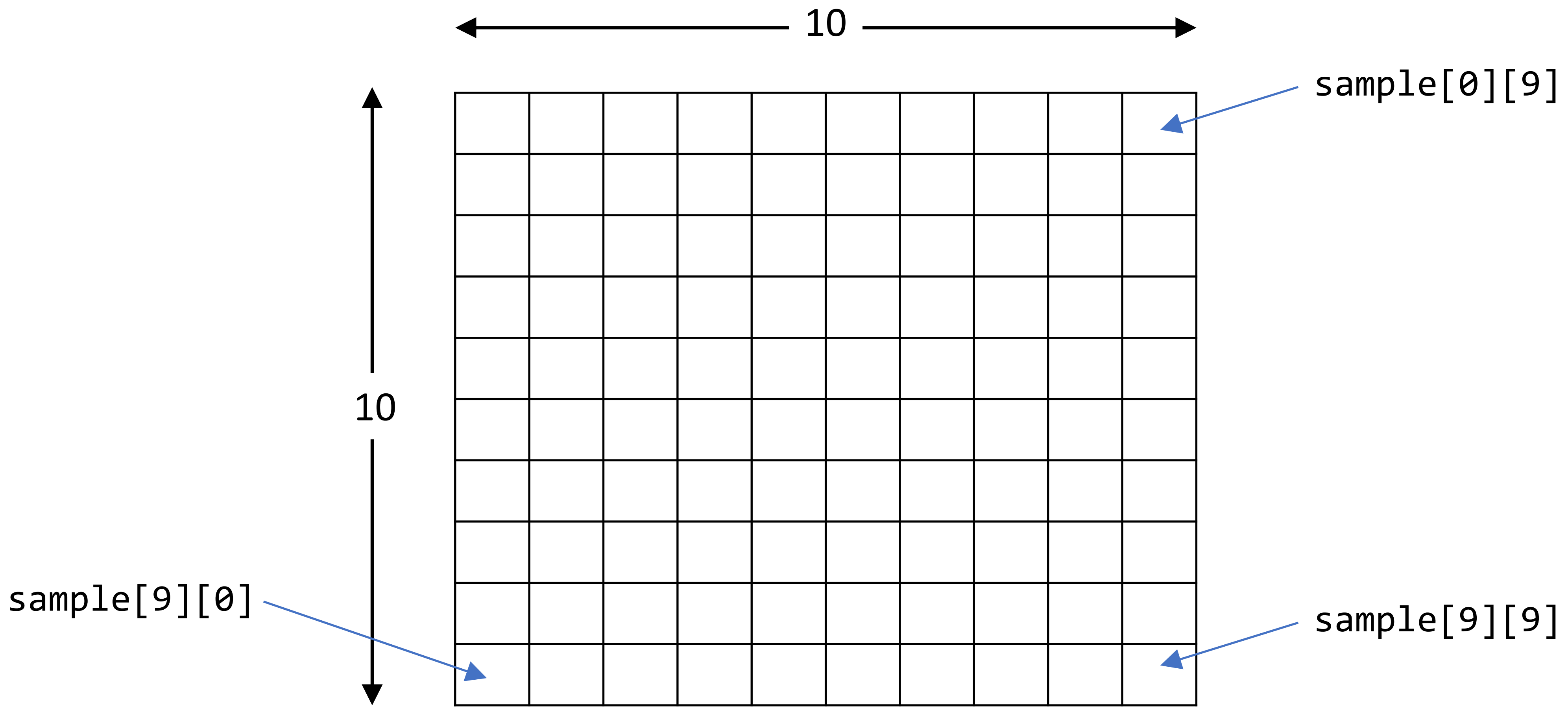
```
type variable_name[row_size][column_size]
```

- **type**: base type of the array, determines the data type of each element in the array
- **row_size**: how many rows the 2D array will hold
- **column_size**: how many column the 2D array will hold
- **variable_name**: the name of the array

Two-Dimensional Arrays

Examples:

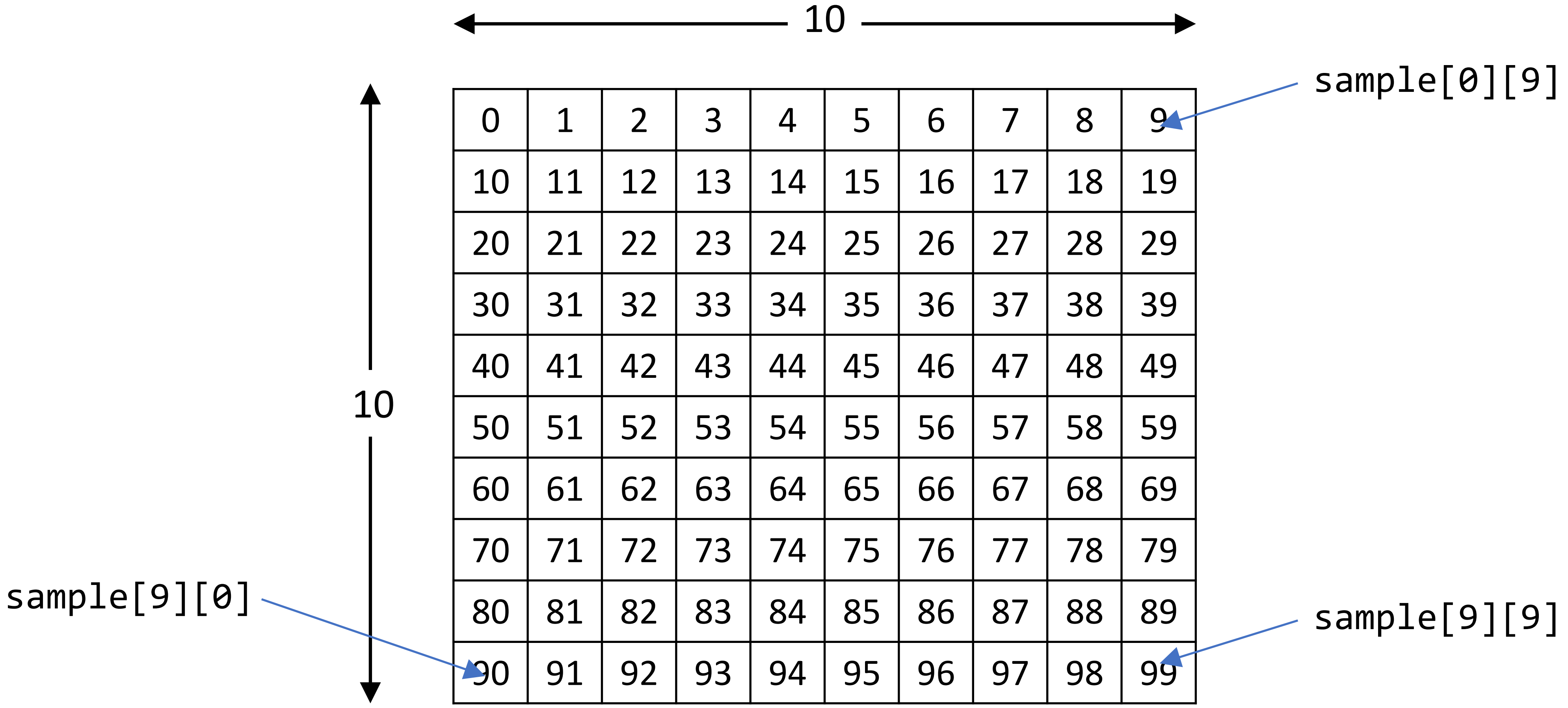
```
int sample[10][10];
```



Two-Dimensional Arrays – Initialization

Examples:

```c
int sample[10][10];
```



0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	64	65	66	67	68	69
70	71	72	73	74	75	76	77	78	79
80	81	82	83	84	85	86	87	88	89
90	91	92	93	94	95	96	97	98	99

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{













}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];









}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];
    for (int i = 0; i < 10; i++)
    {







    }
    return(0);
}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {


        }

    }
    return(0);
}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            sample[i][j] =  ******** ;

        }

    }
    return(0);
}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            sample[i][j] = (i*10)+j;

        }

    }
    return(0);
}
```

Two-Dimensional Arrays – Initialization

Examples:

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{

    int sample[10][10];
    for (int i = 0; i < 10; i++)
    {
        for (int j = 0; j < 10; j++)
        {
            sample[i][j] = (i*10)+j;
            cout << sample[i][j] << " ";
        }
        cout << endl;
    }
    return(0);
}
```

Two-Dimensional Arrays – Intialization

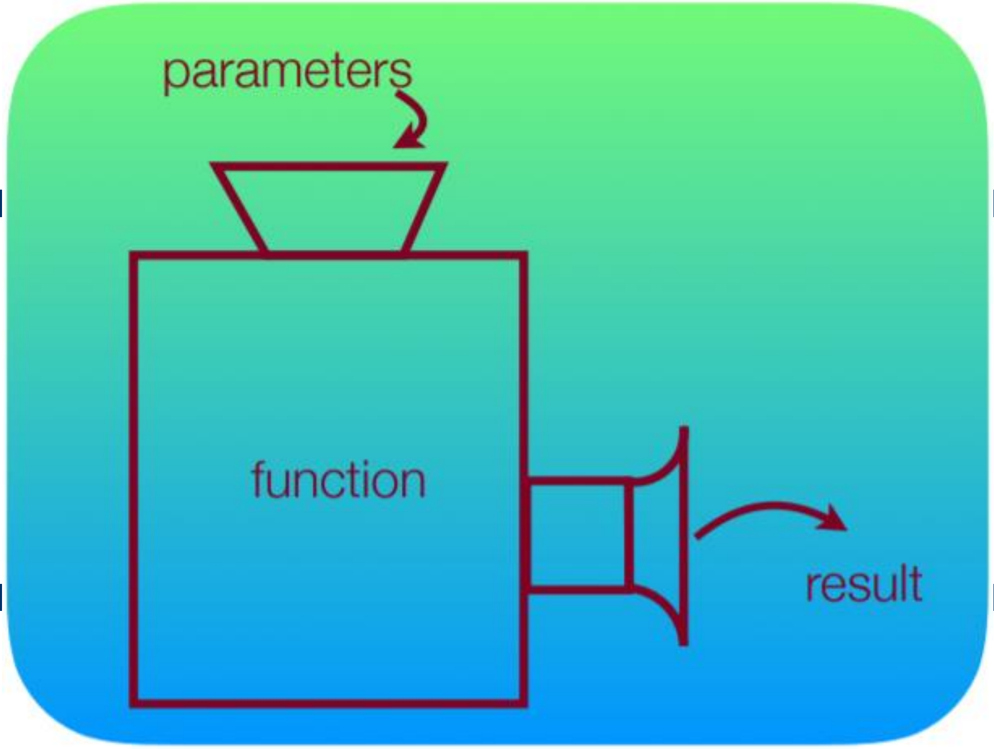
- Multi-dimensional arrays are initialized the same way as one-dimensional arrays.
- For example, the following code fragment initializes an array ***Square*** with the numbers 1 through 4:

```
int Square[2][2] = { 1,2,3,4 };
```

- For better readability, especially for multi-dimensional arrays, one can use sub-aggregate grouping by adding braces accordingly.
- The same declaration as above can also be written as:

```
int Square[2][2] = { {1,2},{3,4} };
```

Functions

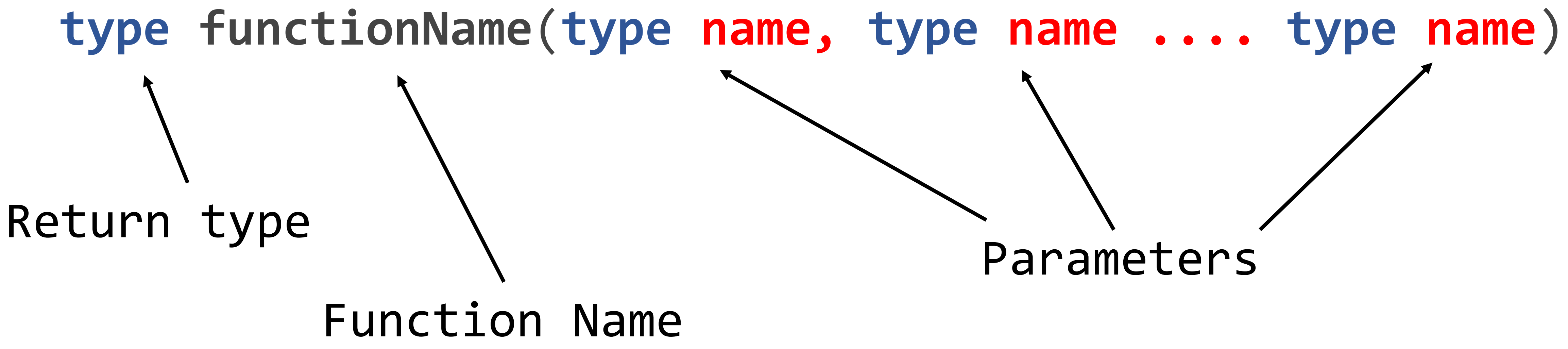


Functions

Definition:

- A function is a block of code which only runs when it is called.
- The general form of a function is:

```
type functionName(type name, type name .... type name)
```



Function – Declaration and Definition

A C++ function consist of two parts: **Declaration** and **Definition**

```
type functionName(type name, type name .... type name)
{
    statement;
    statement;
    statement;
    .
    .
    .
    statement;

    return type;
}
```

Function Declaration

Function Definition

Function - Calling

- Declared functions are not executed immediately.
- They are "**saved for later use**", and will be executed later, when they are called.

To call a function, write the **function's name** followed by two parentheses **()**. Inside the parentheses add **values for the parameters** and in the end a semicolon **;**

```
int main()
{

     functionName(value, value .... value);

 }
```

Function - Advantages

The use of functions in a program allows:

- a program to be broken into small tasks.
- a program to be written and debugged one small part at a time.
- several programmers to work on the same program.
- make a program much easier to read, test and debug.

Function - Example

Function which returns sum of two values:

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

float Sum(float a, float b)      Function Declaration
{
    float result;
    result = a + b;               Function Definition
    return result;
}

int main()
{
    float num1 = 10;
    float num2 = 20;
    float answer = Sum(num1, num2);   Function Calling
    cout << answer;

    return 0;
}
```

Remember: Function declaration should come before function calling

Note: If a function, such as **Sum(float a, float b)** is declared after the main() function, an error will occur. It is because C++ works from top to bottom; which means that if the function is not declared above main(), the program is unaware of it

Function - Example

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

float Sum(float, float);                  Function Declaration

int main()
{
    float num1 = 10;
    float num2 = 20;
    float answer = Sum(num1, num2);           Function Calling
    cout << answer;

    return 0;
}

float Sum(float a, float b)
{
    float result;                         Function Definition
    result = a + b;
    return result;
}
```

Function - Exercise

Write a function that receives three integers and returns the largest of the three. Assume the integers are not equal to one another.

Write a function that receives a character and returns true if the character is a vowel and false otherwise. For this example, vowels include the characters 'a', 'e', 'i', 'o', and 'u'.

Write a function that receives one integer and returns its square root.

Thanks a lot

If you are taking a Nap, **wake up**.......Lecture Over