Modular Programming with Functions

B. Ramamurthy

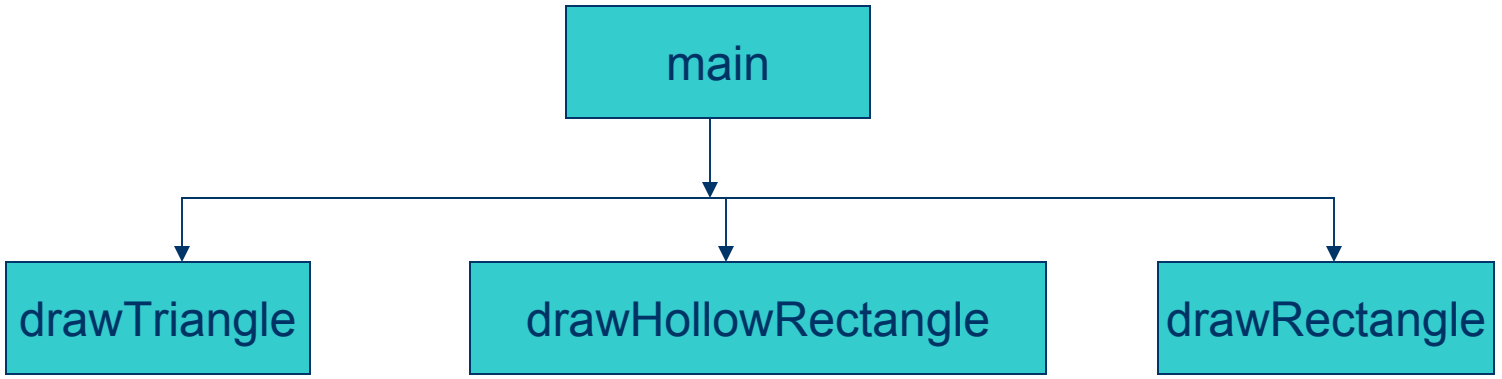
Chapter 5

Introduction

- Complex problems: Divide and conquer
- Reusability of code
- Modular design of software to enhance readability and maintenance.
- Abstraction
- Information hiding

Structure Charts

- Structure charts are often used to represent the modular design.
- Here is an example of a structure chart for StickFigure.



Libraries of Functions

- We have used functions from many libraries:
- cmath : pow, sin, cos
- fstream : open, close, get, set
- iomanip: setw, setprecision
- iostream : << , >>
- Programmers can define functions for use in their program: programmer-defined functions.

Functions Definition

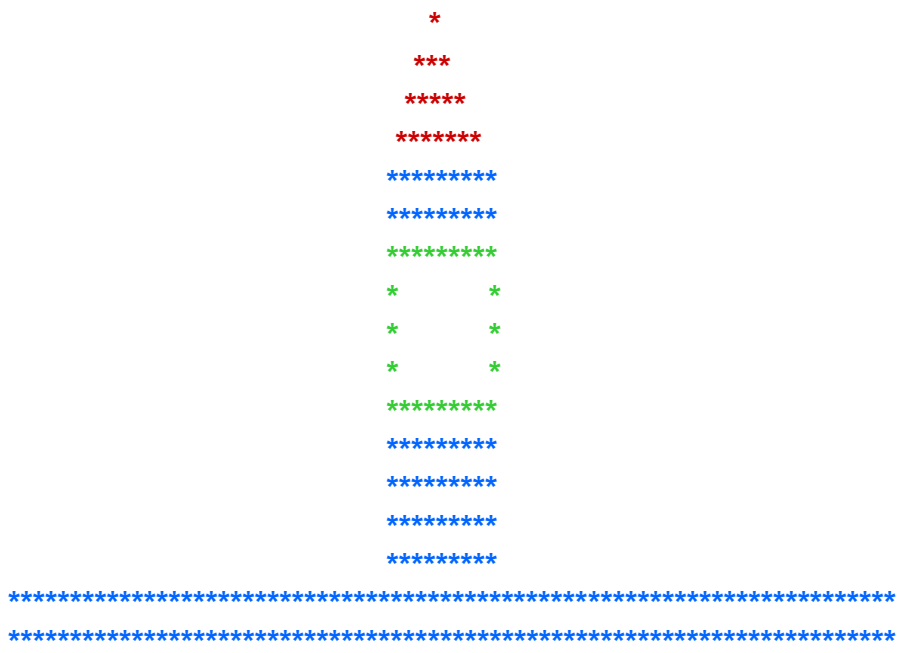
1. Function header
2. Function parameters
3. Function body
4. Function return value

- In fact

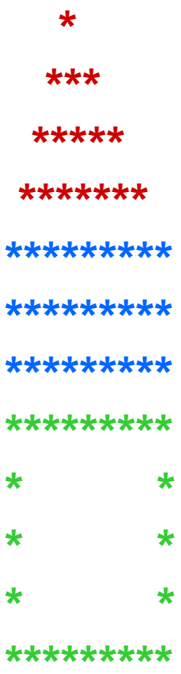
```
int main()
 {
  …
  return 0;
}
```

is a function.

Example: StickFigure



Part of Stick Figure



Step1 : Draw triangle
Step2: Draw rectangle
Step3: Draw hollow rectangle

drawTriangle(..)

```
for (int i3 = 0; i3 < row; i3++)
        {   for (int k = 1; k<= offset; k++)
                        cout<< ' ';
                for (int j = 1; j <= col; j++ )
                        cout <<'*';
                offset = offset -1;
                col = col +2;
                cout<< endl;
        }
```

drawRectangle(..)

```
for (int i4 = 1; i4 <= length; i4++)
        {   for (int k = 1; k<= offset; k++)
                                cout<< ' ';
                    for (int j = 1; j<= width; j++)
                                cout <<'*';
                    cout<< endl;
        }
```

drawHollowRectangle(..)

```
for (int i4 = 1; i4 <= length; i4++)
        {   for (int k = 1; k<= offset; k++) cout<< '  ';
                if ((i4 ==1 ) || (i4 == length))
                { for (int j = 1; j<= width; j++)
                        cout <<'*';
                }
                else
                {
                        cout << '*';
                        for (int k =1; k<= width-2; k++)
                                cout << ‘  ';
                        cout << '*';
                }
                cout<< endl;
        }
```

Function syntax

```
type functioName (parameters)
{
  declare local variables/constants
  statements
}
```

Example

```
int factorial (int n)
{
    int prod;
    for (int i = 1; i <= n; i++)
       prod = prod *i;
    return prod;
}
```