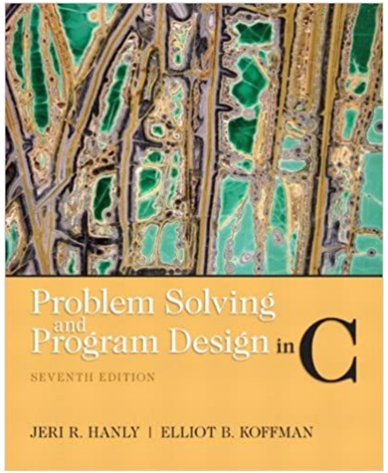
Faculty of Engineering and Technology
Department of Computer Science


Problem Solving and Program Design in C
SEVENTH EDITION
JERI R. HANLY | ELLIOT B. KOFFMAN

Introduction to Computers and Programming (Comp 133)

References :
Book : Problem Solving and Program Design in C (7th Edition) 7th Edition
Slides : Dr. Radi Jarrar , Dr. Abdallah Karakra , Dr. Majdi Mafarja.

Top-Down Design with Functions

Chapter 3

Functions

- A top-down design is the decomposition of a system into smaller parts in order to comprehend its compositional sub-systems

- In programming, a function is a segment that groups a set of code statements in a given order and that can be referenced by a unique name to perform a specific task.

- A C program has at least one function **main()**. Without main() function, there is technically no C program

Chapter 3

- Types of C functions
  - Library function
  - User defined function

Library function

- A primary goal of predefined functions is **code reuse**.
- C support many library that embedded predefined functions.
  - mathematical computations <math.h>

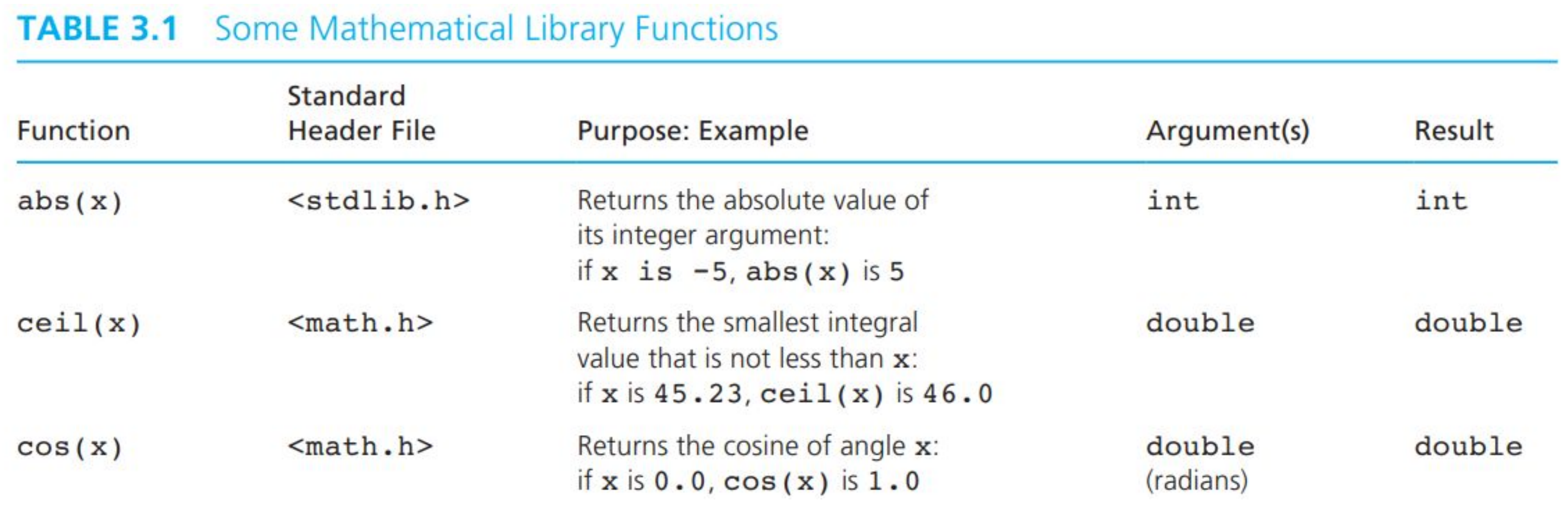


TABLE 3.1 Some Mathematical Library Functions

Function	Standard Header File	Purpose: Example	Argument(s)	Result
`abs(x)`	`<stdlib.h>`	Returns the absolute value of its integer argument: if `x is -5`, `abs(x)` is `5`	`int`	`int`
`ceil(x)`	`<math.h>`	Returns the smallest integral value that is not less than `x`: if `x` is `45.23`, `ceil(x)` is `46.0`	`double`	`double`
`cos(x)`	`<math.h>`	Returns the cosine of angle `x`: if `x` is `0.0`, `cos(x)` is `1.0`	`double` (radians)	`double`

Library function

`exp(x)`	`<math.h>`	Returns e^x where e = `2.71828...`: if `x` is `1.0`, `exp(x)` is `2.71828`	`double`	`double`
`fabs(x)`	`<math.h>`	Returns the absolute value of its type `double` argument: if `x` is `-8.432`, `fabs(x)` is `8.432`	`double`	`double`
`floor(x)`	`<math.h>`	Returns the largest integral value that is not greater than `x`: if `x` is `45.23`, `floor(x)` is `45.0`	`double`	`double`
`log(x)`	`<math.h>`	Returns the natural logarithm of `x` for `x > 0.0`: if `x` is `2.71828`, `log(x)` is `1.0`	`double`	`double`
`log10(x)`	`<math.h>`	Returns the base-10 logarithm of `x` for `x > 0.0`: if `x` is `100.0`, `log10(x)` is `2.0`	`double`	`double`
`pow(x, y)`	`<math.h>`	Returns x^y. If `x` is negative, `y` must be integral: if `x` is `0.16` and `y` is `0.5`, `pow(x,y)` is `0.4`	`double`, `double`	`double`
`sin(x)`	`<math.h>`	Returns the sine of angle `x`: if `x` is `1.5708`, `sin(x)` is `1.0`	`double` (radians)	`double`
`sqrt(x)`	`<math.h>`	Returns the nonnegative square root of `x` ($\sqrt{x}$) for $x \geq 0.0$: if `x` is `2.25`, `sqrt(x)` is `1.5`	`double`	`double`
`tan(x)`	`<math.h>`	Returns the tangent of angle `x`: if `x` is `0.0`, `tan(x)` is `0.0`	`double` (radians)	`double`

Library function example

```
 #include <stdio.h>
#include <math.h>
int main(){
 float num,root;
 printf("Enter a number to find square root.");
 scanf("%f",&num);
 root=sqrt(num);/* Computes the square root of num and stores in root. */
 printf("Square root of %.2f=%.2f",num,root);
 return 0;
}
```

```
Enter a number to find square root.12
Square root of 12.00=3.46
```

Library function example

```
#include <stdio.h>

#include <math.h>

int main ()

{

 printf("Value 8.0 ^ 3 = %lf\n", pow(8.0, 3));

 printf("Value 3.05 ^ 1.98 = %lf", pow(3.05, 1.98));

 return(0);

}
```

```
Value 8.0 ^ 3 = 512.000000
Value 3.05 ^ 1.98 = 9.097324
```

Chapter 3

- Types of C functions
  - Library function
  - User defined function

User defined functions

Why Functions ?

- Divide the programs into separate functions (instead of big “chunk”'). This make it easy to debug the code and handling error.

- **Reusability** :
  - Defined function can be used over and over and over again.
  - Invoke(call) the same function many times in the program.
  - Use same function in several different (and separate) programs.

Types of functions

1. Function with no arguments and no return value.
2. Function with no arguments but return value
3. Function with arguments and no return value
4. Function with argument and a return value

```
Syntax : [return_type]  [void] function_name ( [ parameter_list ] )
{
body of function ;
Return [value];
}
```

User defined functions

Function with no arguments and no return value.

```
void sum ( );

void sum ( )

{ int result,x,y;

 scanf("%d%d",&x,&y);

 result= x+y;

 printf("The result= %d",result);

}

int main() {

sum ( ) ;

return 0; }
```

To write a function:

Function prototype

Function Definition

Function Call

User defined functions

Function with no arguments but return value

```
int sum ( );
int main() {
int ResultSum=sum ( ) ;
 printf("The result= %d",ResultSum);
return 0;
}
int sum ( )
{ int result , x , y;
 scanf("%d%d",&x,&y);
 result= x+y;
 return result
}
```

To write a function:

Function prototype

Function Definition

Function Call

User defined functions

Function with arguments and no return value

```
void sum (int,int );

int main() {

sum ( 5,6) ;

return 0;

}

void sum (int x, int y )

{ int result;

 result= x+y;

 printf("The result= %d",result);

}
```

To write a function:

Function prototype

Function Definition

Function Call

User defined functions

Function with argument and a return value

```
int sum (int , int );
int main() {
int R= sum ( 5,6) ;
printf("The result= %d",R);
return 0;
}
int sum (int x, int y )
{ int result;
 result= x+y;
return result;

}
```

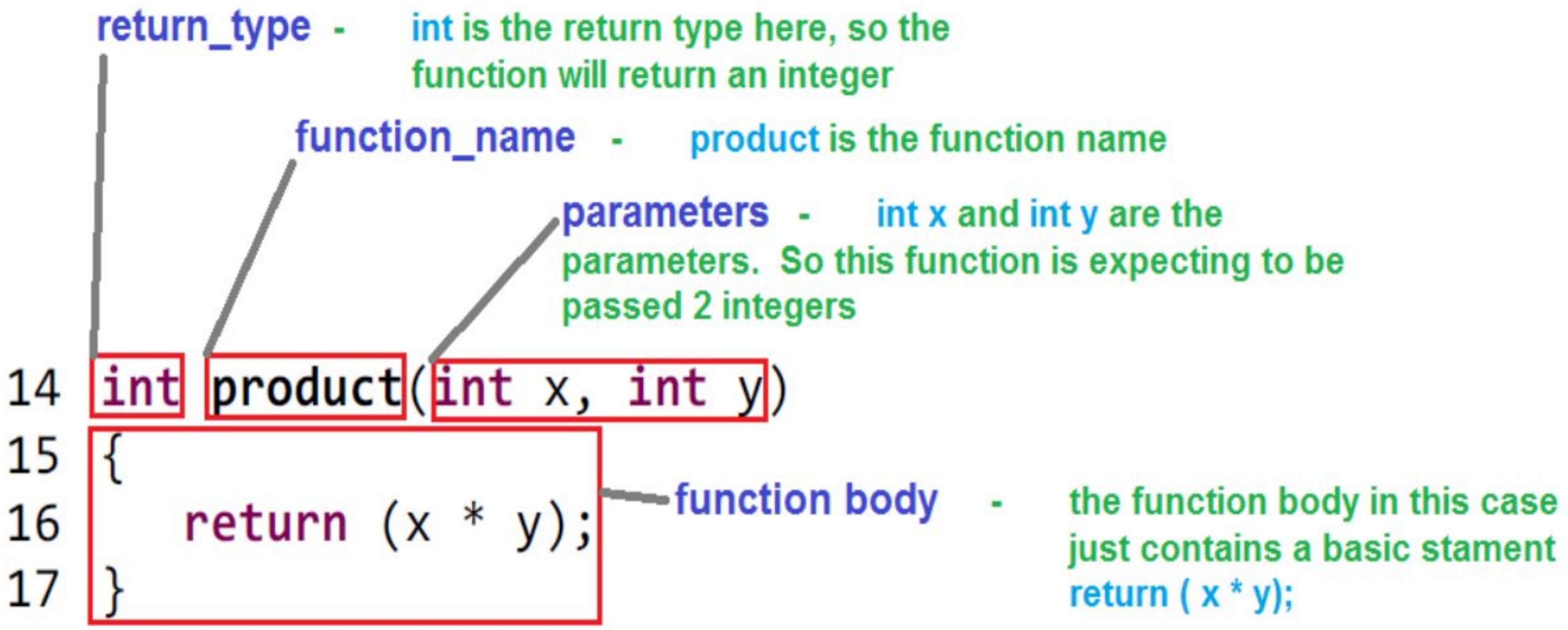
To write a function:

Function prototype

Function Definition

Function Call

User defined functions



return_type - int is the return type here, so the function will return an integer

function_name - product is the function name

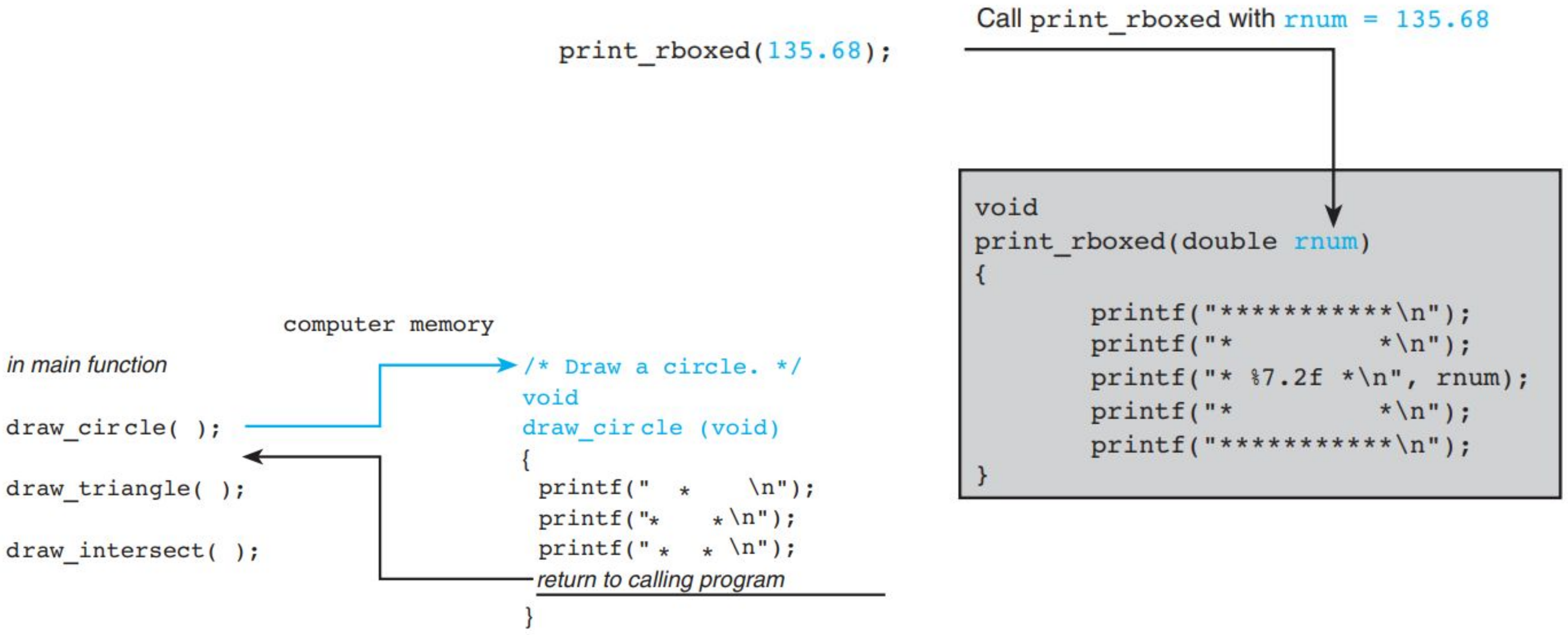
parameters - int x and int y are the parameters. So this function is expecting to be passed 2 integers

function body - the function body in this case just contains a basic stament return (x * y);

```
int product(int x, int y)
{
    return (x * y);
}
```

User defined functions

Flow of Control Between the main Function and a Function Subprogram


Call print_rboxed with rnum = 135.68
computer memory
in main function
return to calling program

```
print_rboxed(135.68);
```

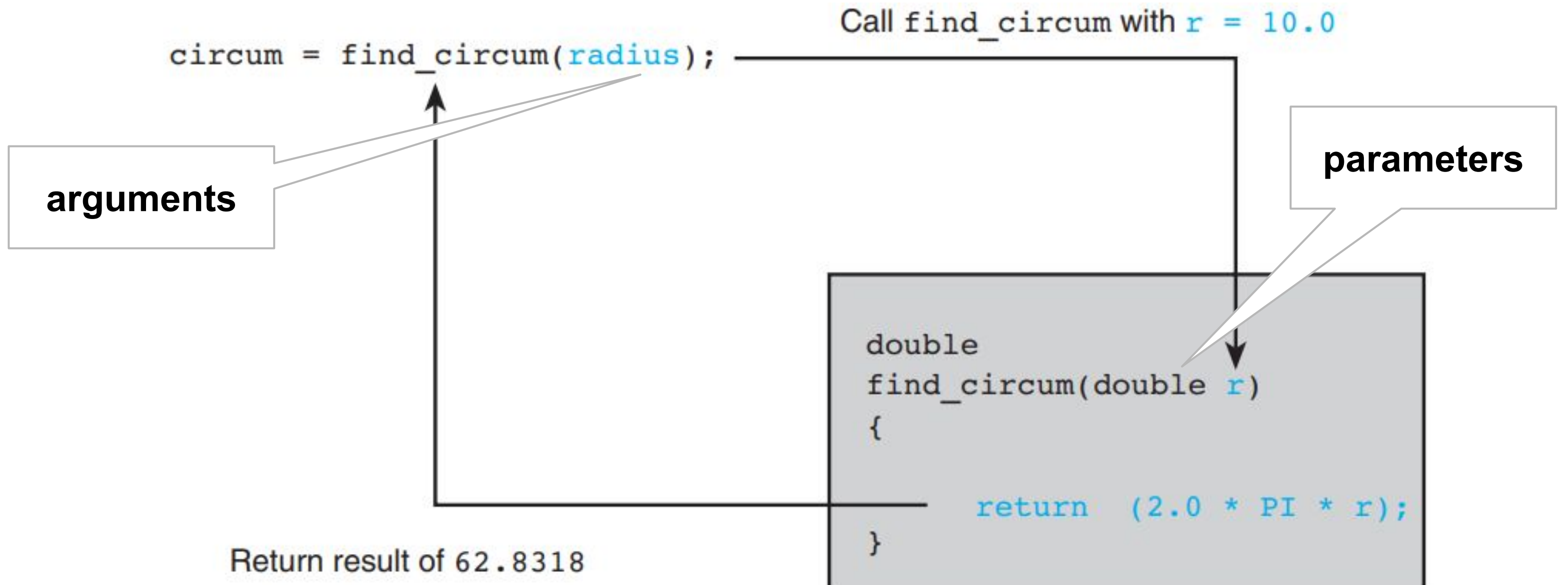
```
void
print_rboxed(double rnum)
{
        printf("***********\n");
        printf("*          *\n");
        printf("* %7.2f *\n", rnum);
        printf("*          *\n");
        printf("***********\n");
}
```

```
draw_circle( );

draw_triangle( );

draw_intersect( );
```

```
/* Draw a circle. */
void
draw_circle (void)
{
 printf("  *   \n");
 printf("*    *\n");
 printf(" *  * \n");
}
```

User defined functions

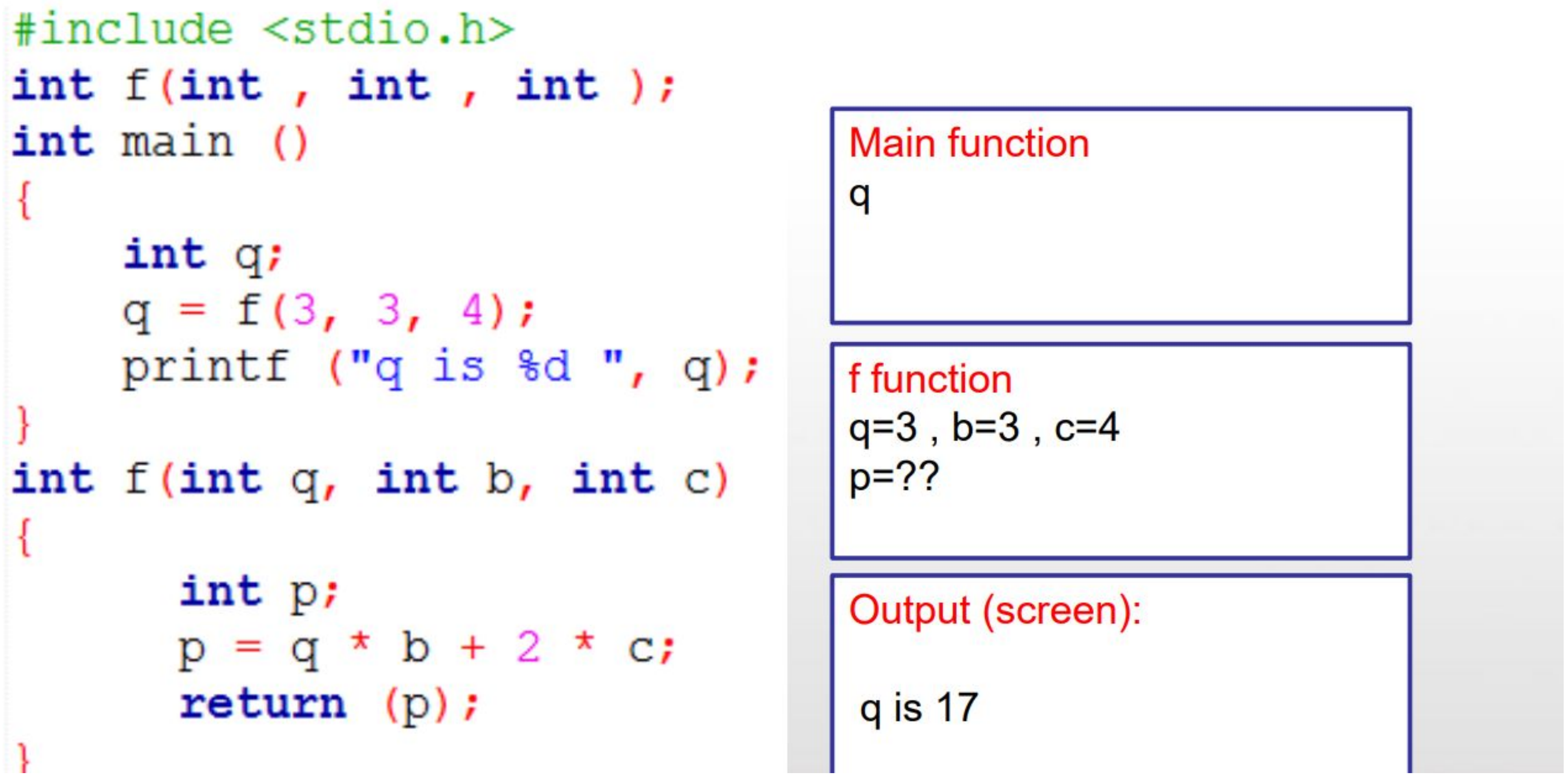
Flow of Control Between the main Function and a Function Subprogram



```
circum = find_circum(radius);
```

```
double
find_circum(double r)
{

    return  (2.0 * PI * r);
}
```

User defined functions example



```c
#include <stdio.h>
int f(int , int , int );
int main ()
{
    int q;
    q = f(3, 3, 4);
    printf ("q is %d ", q);
}
int f(int q, int b, int c)
{
      int p;
      p = q * b + 2 * c;
      return (p);
}
```

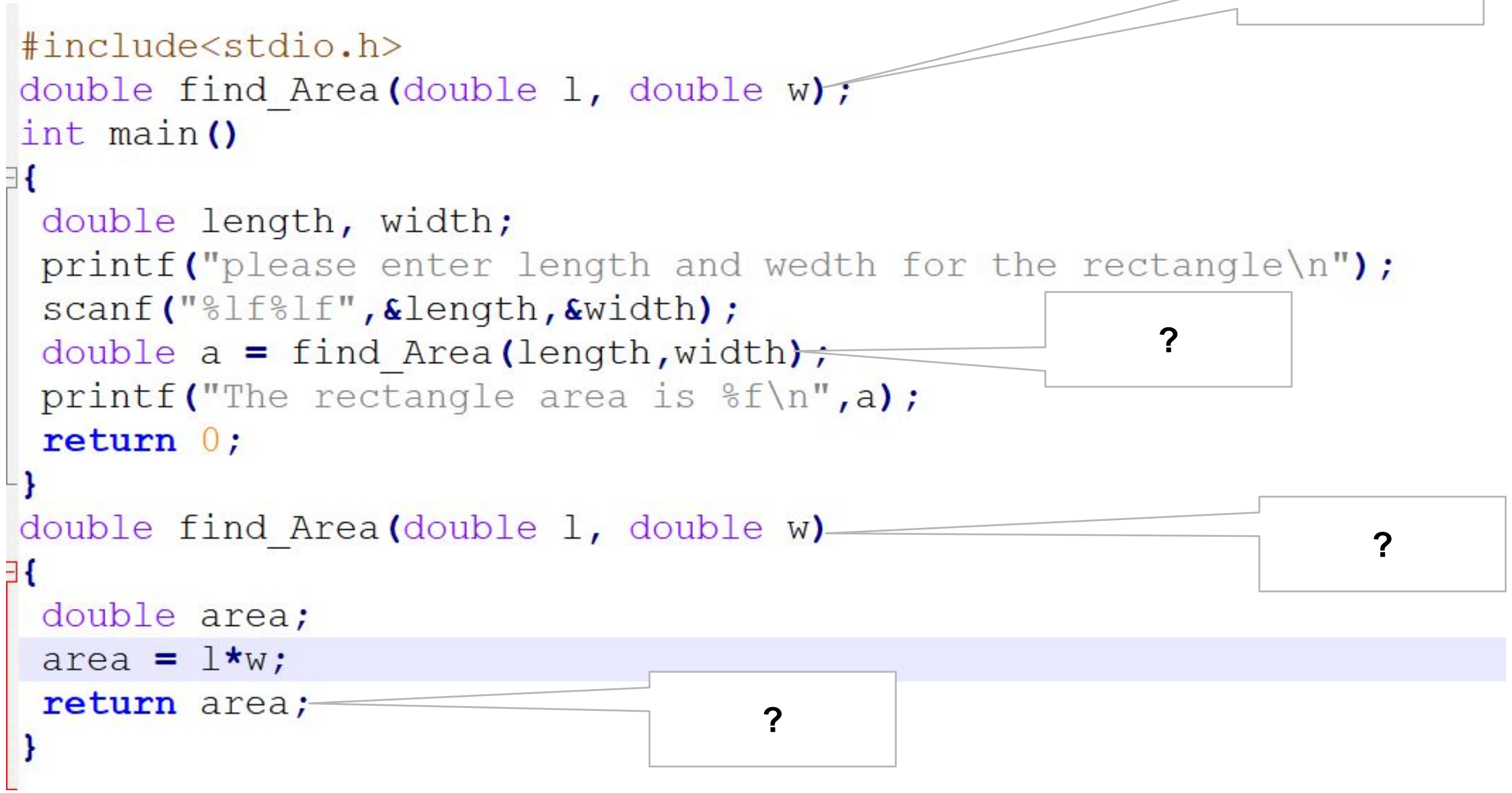
Main function
q

f function
q=3 , b=3 , c=4
p=??

Output (screen):

```
q is 17
```

User defined functions practice



```c
#include<stdio.h>
double find_Area(double l, double w);
int main()
{
 double length, width;
 printf("please enter length and wedth for the rectangle\n");
 scanf("%lf%lf",&length,&width);
 double a = find_Area(length,width);
 printf("The rectangle area is %f\n",a);
 return 0;
}
double find_Area(double l, double w)
{
 double area;
 area = l*w;
 return area;
}
```

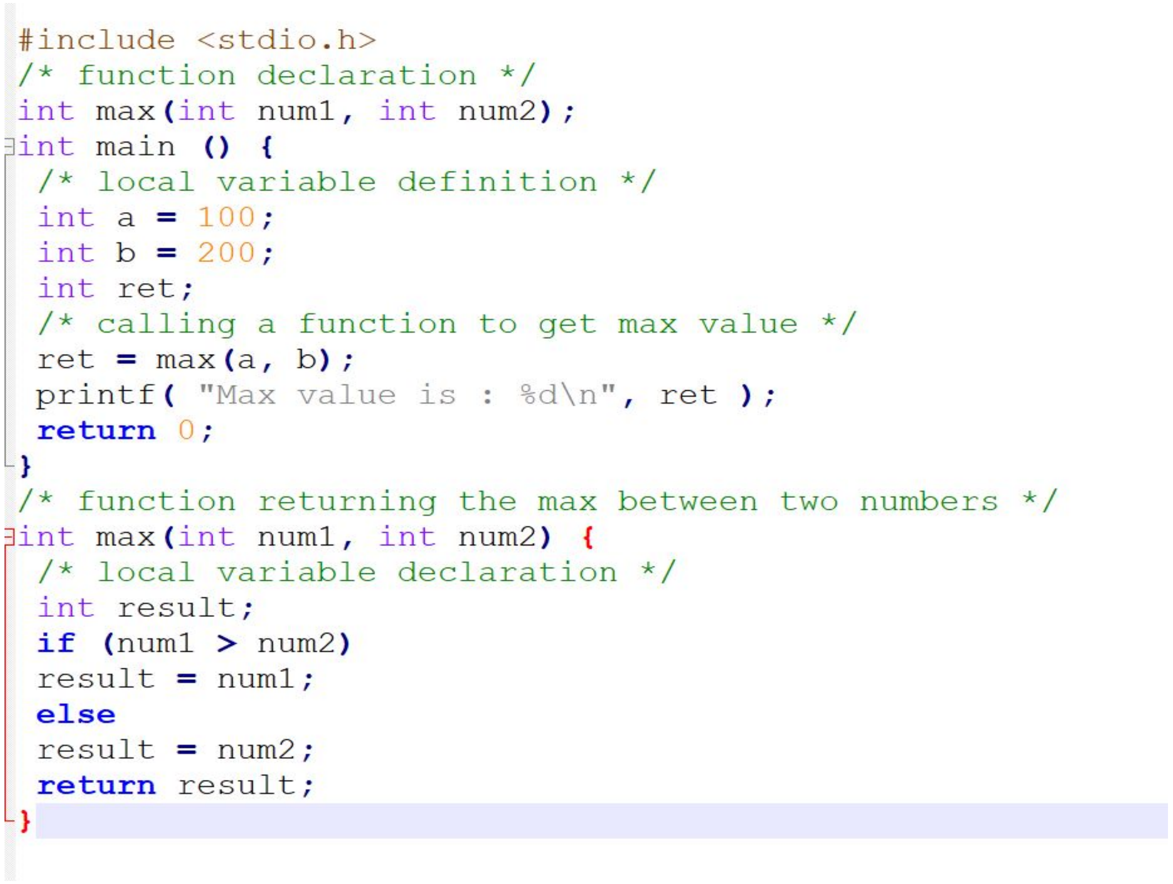
?

?

?

?

User defined functions practice



```c
#include <stdio.h>
/* function declaration */
int max(int num1, int num2);
int main () {
 /* local variable definition */
 int a = 100;
 int b = 200;
 int ret;
 /* calling a function to get max value */
 ret = max(a, b);
 printf( "Max value is : %d\n", ret );
 return 0;
}
/* function returning the max between two numbers */
int max(int num1, int num2) {
 /* local variable declaration */
 int result;
 if (num1 > num2)
 result = num1;
 else
 result = num2;
 return result;
}
```

User defined functions practice

Write a complete c program to do the following.

Y= x3 + x2 + x

Your program should include two functions, cubic to return x to the power of three and square to return x to the power of two

User defined functions Extra Exercises

- Which of the following is a correct function definition?
  1. int funct();
  2. int funct(int x) {return x=x+1;}
  3. void funct(int) {printf("Hello");}
  4. void funct(x) {printf("Hello")}
- Which of the following is a valid function call (assuming the function exists)?
  1. funct;
  2. funct x, y;
  3. funct();
  4. int funct();

User defined functions Extra Exercises

- When using a function, what is the first thing you must do?

  a. prototype b. declare c. initialize

- Where should the prototype be?

  a. after int main() b. before int main() c. a prototype isn't necessary

- Say we have a function, double subtract (double x, double y), what is the correct way to call this function in the main program?

  a. subtract (x) b. subtract (y) c. subtract (x,y)

- Write a function to return the square of an integer number ?

Thank You.

جامعة بيرزيت
BIRZEIT UNIVERSITY