Hacettepe University

Computer Engineering Department

Programming in

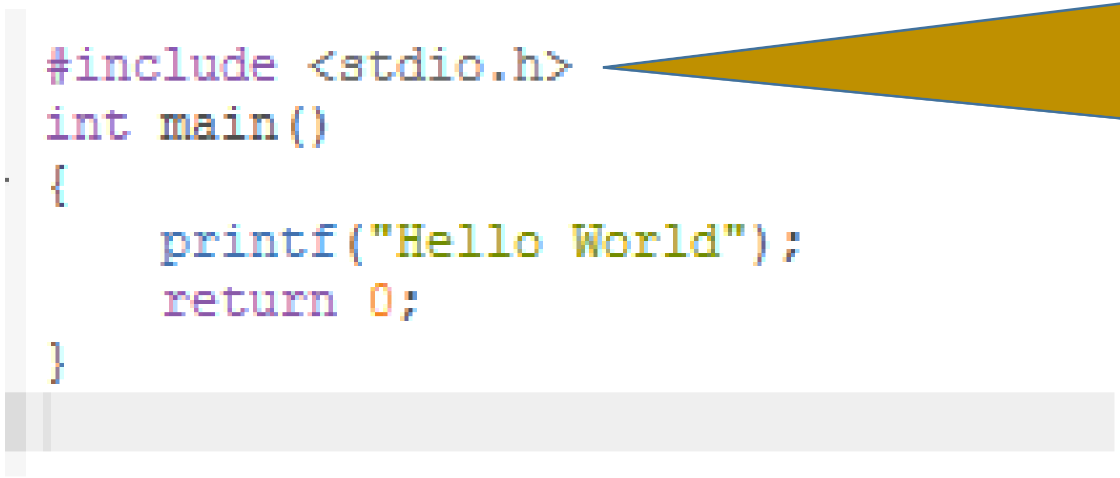
python & {C}

BBM103 Introduction to Programming Lab 1

Week 12

Fall 2017

First Program with C



```c
#include <stdio.h>
int main()
{
    printf("Hello World");
    return 0;
}
```

The **#include <stdio.h>** is a preprocessor command.

This command tells compiler to include the contents of stdio.h (standard input and output) file in the program.

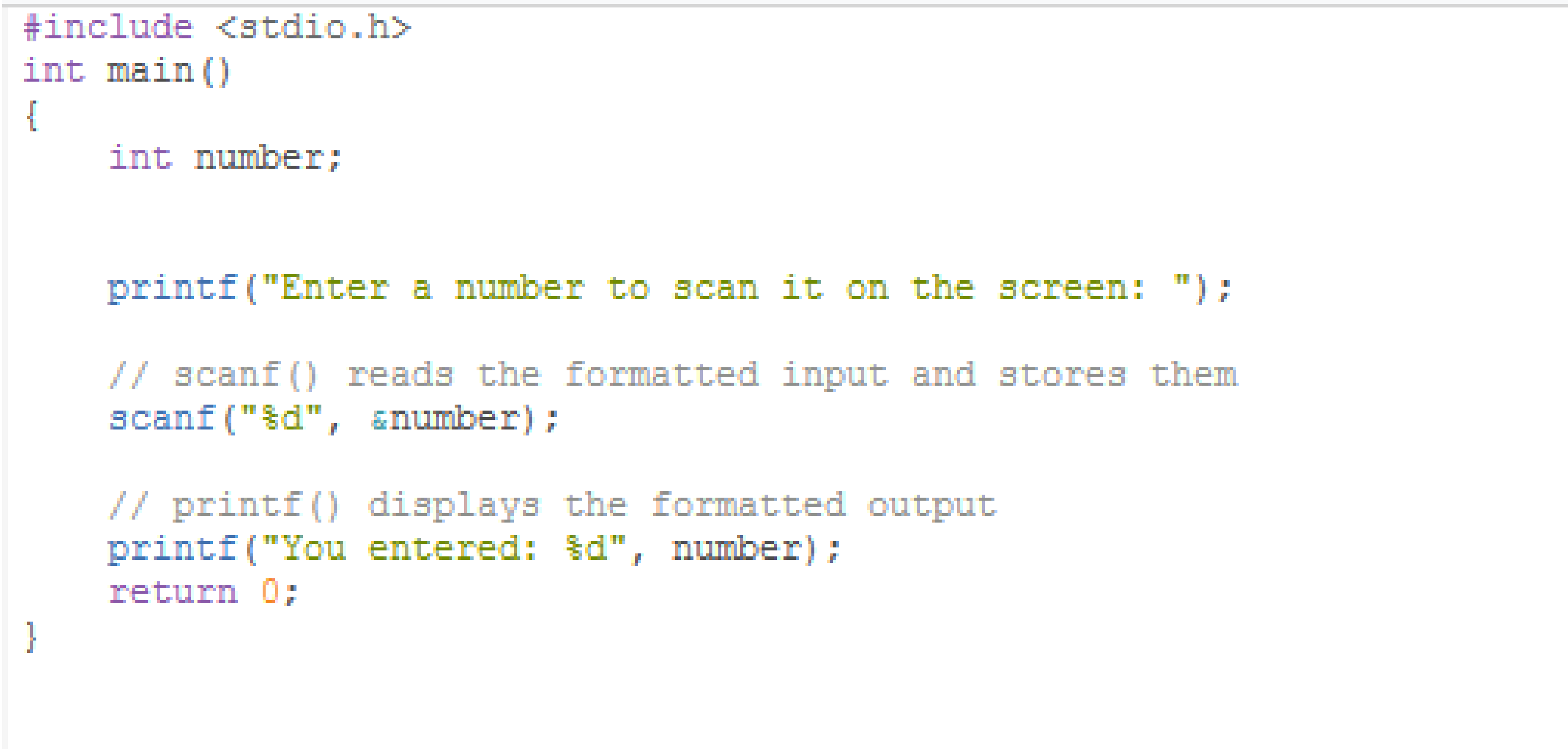
The stdio.h file contains functions such as scanf() and print() to take input and display output respectively.

- The execution of a C program starts from the main() function

- The printf() is a library function to send formatted output to the screen. In this program, the printf() displays *Hello, World!* text on the screen.

- The return 0; statement is the "Exit status" of the program. In simple terms, program ends with this statement.

printf() and scanf()

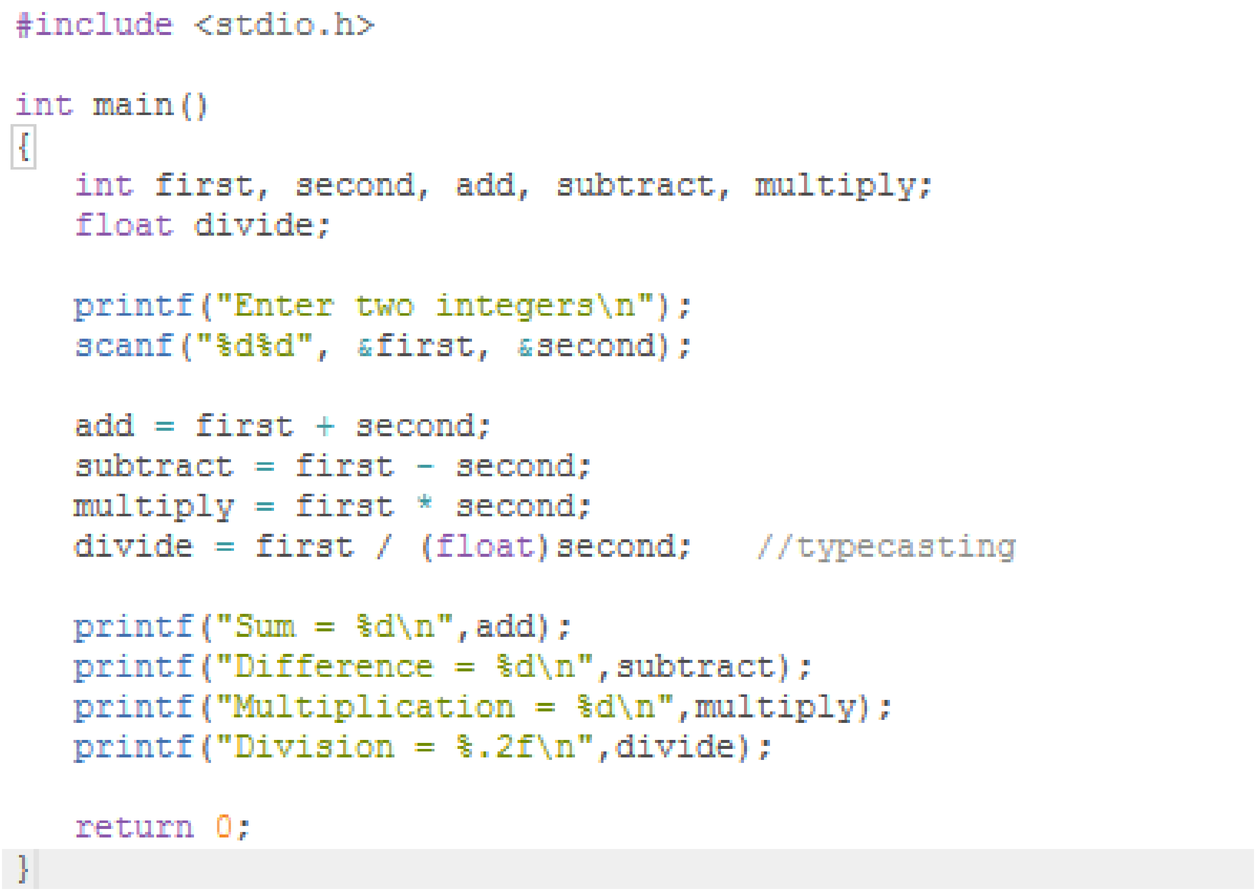


```c
#include <stdio.h>
int main()
{
    int number;


    printf("Enter a number to scan it on the screen: ");

    // scanf() reads the formatted input and stores them
    scanf("%d", &number);

    // printf() displays the formatted output
    printf("You entered: %d", number);
    return 0;
}
```

scanf()

the scanf() function reads an integer data from the user and stores in variable *number*.

Basic Mathematical Operations



```c
#include <stdio.h>

int main()
{
   int first, second, add, subtract, multiply;
   float divide;

   printf("Enter two integers\n");
   scanf("%d%d", &first, &second);

   add = first + second;
   subtract = first - second;
   multiply = first * second;
   divide = first / (float)second;   //typecasting

   printf("Sum = %d\n",add);
   printf("Difference = %d\n",subtract);
   printf("Multiplication = %d\n",multiply);
   printf("Division = %.2f\n",divide);

   return 0;
}
```

In c, as a general rule
integer/integer = integer
float/integer = float
integer/float = float.

So we convert denominator to float in program.
This explicit conversion is known as typecasting.

printf()

Format Specifiers

%i or %d	int
%c	char
%f	float (see also the note below)
%s	string

Format Specifiers

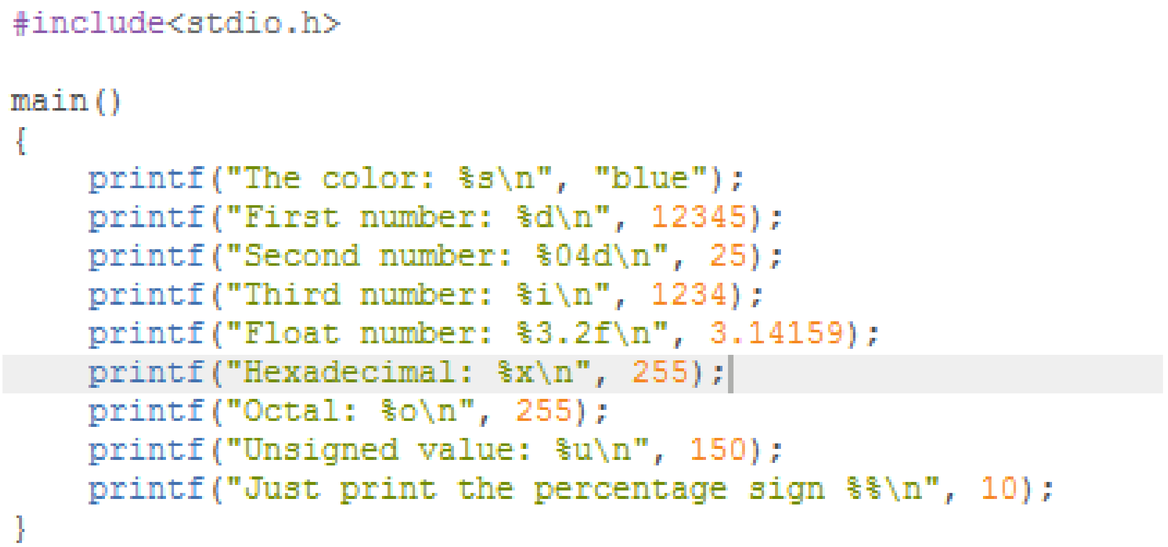
Commonly used escape sequences are:

\n	:newline
\t	:tab
\v	:vertical tab
\f	:new page
\b	:backspace
\r	:carriage return
\n	:newline

Example: Formatting

%d	:print as a decimal integer
%6d	:print as a decimal integer with a width of at least 6 wide
%f	:print as a floating point
%4f	:print as a floating point with a width of at least 4 wide
%.4f	:print as a floating point with a precision of four characters after the decimal point
%3.2f	:print as a floating point at least 3 wide and a precision of 2

Example: Formatting



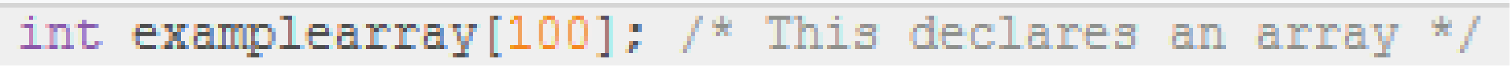
```
#include<stdio.h>

main()
{
    printf("The color: %s\n", "blue");
    printf("First number: %d\n", 12345);
    printf("Second number: %04d\n", 25);
    printf("Third number: %i\n", 1234);
    printf("Float number: %3.2f\n", 3.14159);
    printf("Hexadecimal: %x\n", 255);
    printf("Octal: %o\n", 255);
    printf("Unsigned value: %u\n", 150);
    printf("Just print the percentage sign %%\n", 10);
}
```

Output:

```
The color: blue
First number: 12345
Second number: 0025
Third number: 1234
Float number: 3.14
Hexadecimal: ff
Octal: 377
Unsigned value: 150
Just print the percentage sign %
```

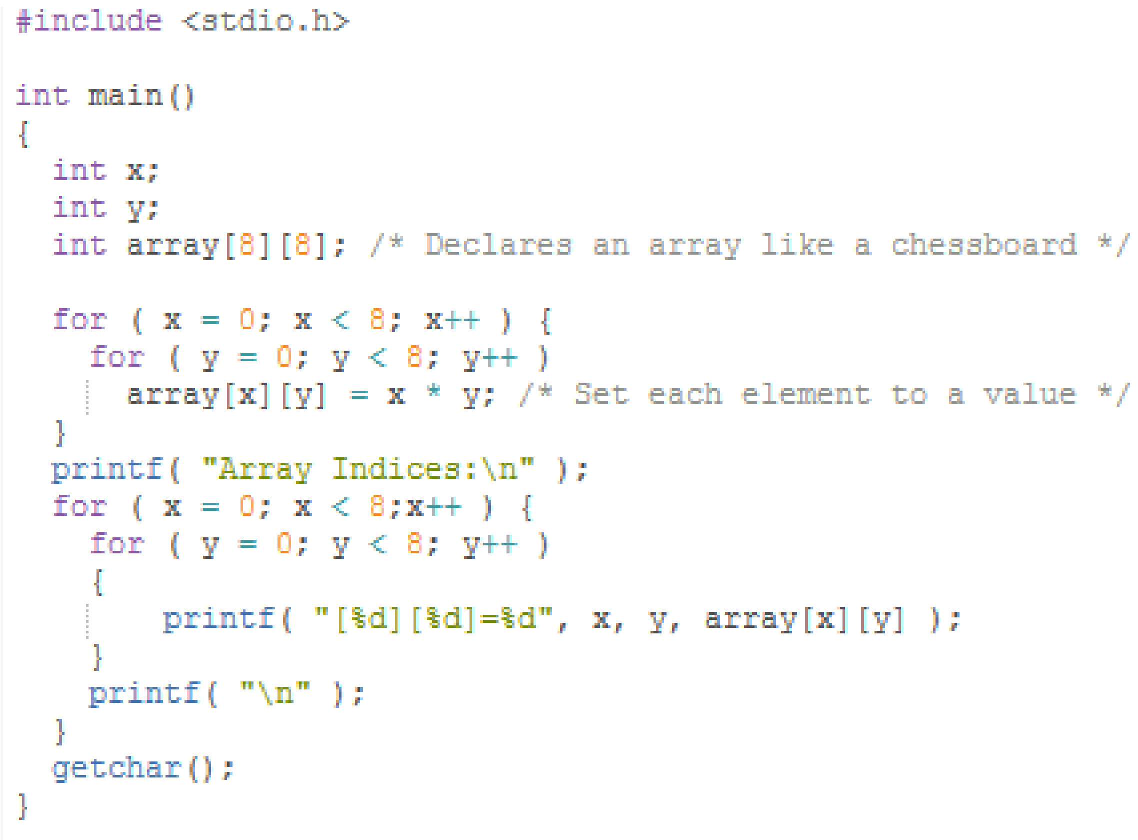
Arrays in C

Defining array



```
int examplearray[100]; /* This declares an array */
```

Example: Array

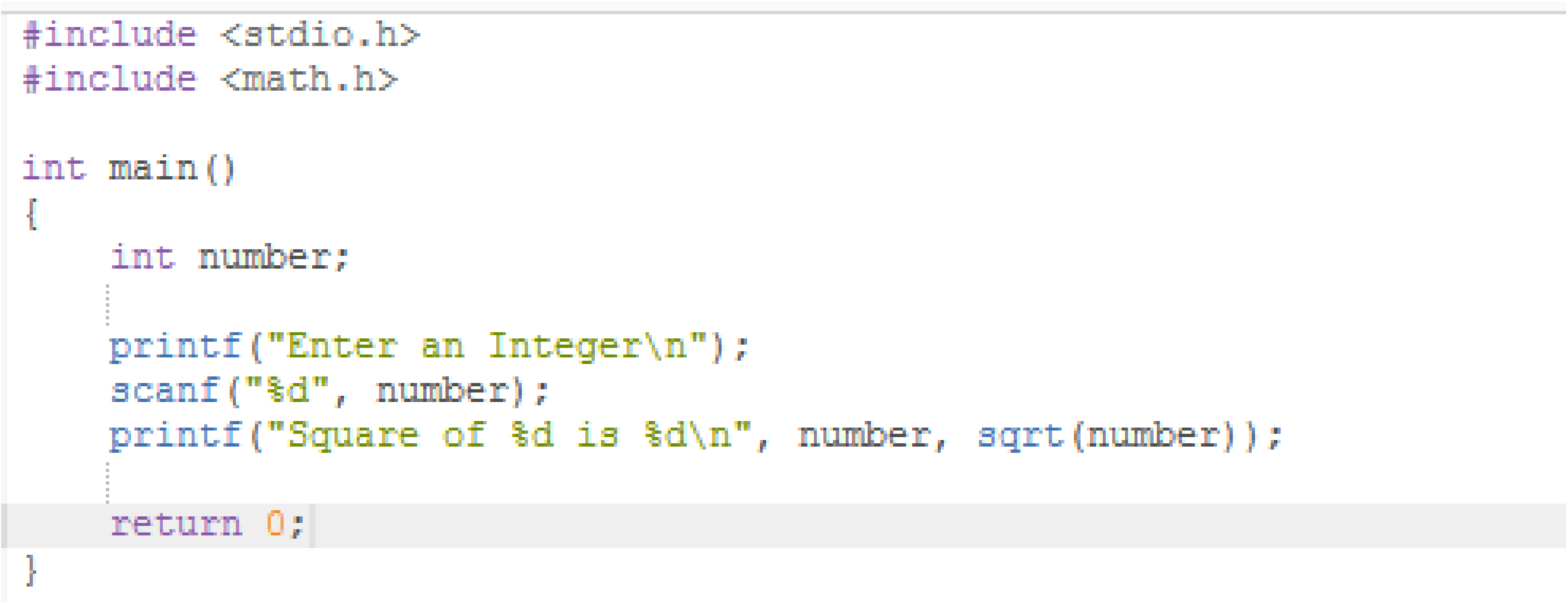


```
#include <stdio.h>

int main()
{
  int x;
  int y;
  int array[8][8]; /* Declares an array like a chessboard */

  for ( x = 0; x < 8; x++ ) {
    for ( y = 0; y < 8; y++ )
      array[x][y] = x * y; /* Set each element to a value */
  }
  printf( "Array Indices:\n" );
  for ( x = 0; x < 8;x++ ) {
    for ( y = 0; y < 8; y++ )
    {
        printf( "[%d][%d]=%d", x, y, array[x][y] );
    }
    printf( "\n" );
  }
  getchar();
}
```

Output:

```
Array Indices:

[0][0]=0
[0][1]=0
[0][2]=0
[0][3]=0
[0][4]=0
[0][5]=0
[0][6]=0
[0][7]=0
[1][0]=0
[1][1]=1
[1][2]=2
[1][3]=3
[1][4]=4
[1][5]=5
[1][6]=6
[1][7]=7
[2][0]=0
[2][1]=2
[2][2]=4
[2][3]=6
[2][4]=8
[2][5]=10
[2][6]=12
[2][7]=14
[3][0]=0
[3][1]=3
[3][2]=6
[3][3]=9
[3][4]=12
[3][5]=15
[3][6]=18
[3][7]=21
[4][0]=0
[4][1]=4
[4][2]=8
[4][3]=12
[4][4]=16
[4][5]=20
[4][6]=24
[4][7]=28
[5][0]=0
[5][1]=5
[5][2]=10
[5][3]=15
[5][4]=20
[5][5]=25
[5][6]=30
[5][7]=35
[6][0]=0
[6][1]=6
[6][2]=12
[6][3]=18
[6][4]=24
[6][5]=30
[6][6]=36
[6][7]=42
[7][0]=0
[7][1]=7
[7][2]=14
[7][3]=21
[7][4]=28
[7][5]=35
[7][6]=42
[7][7]=49
```

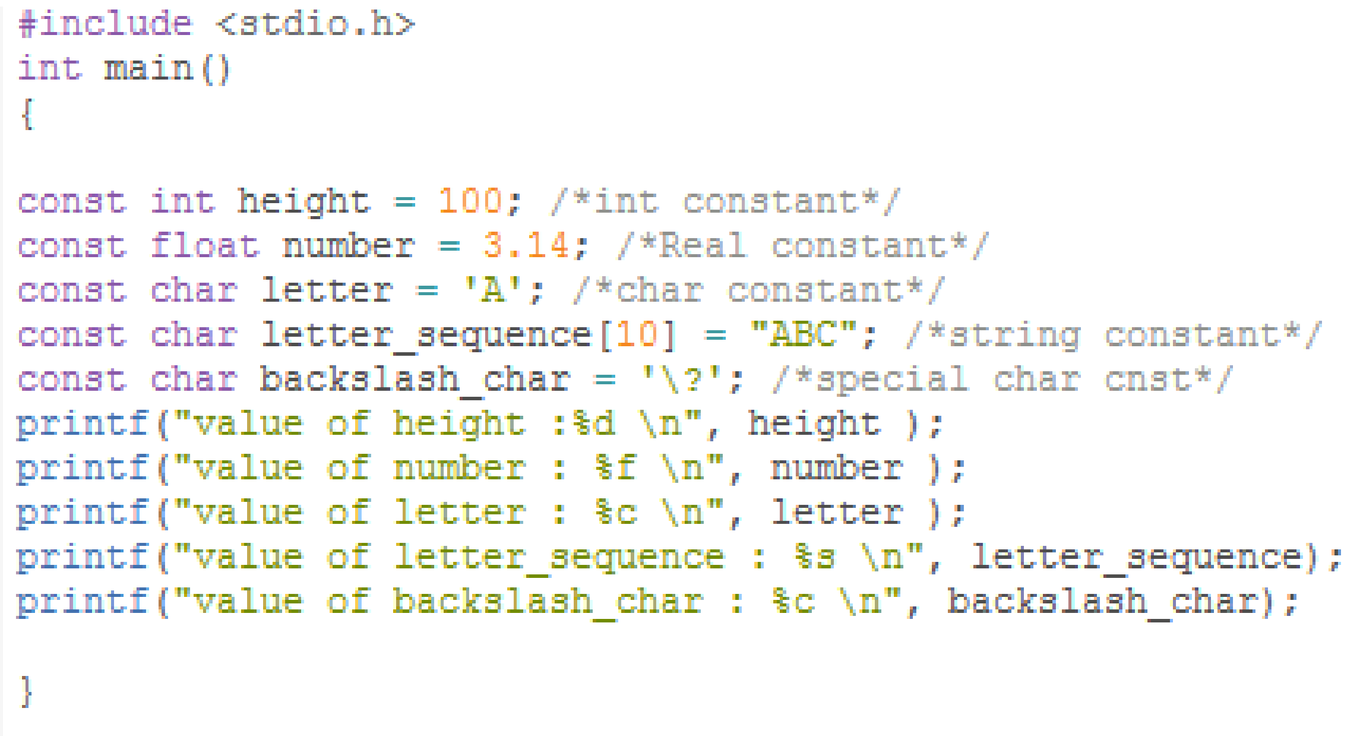
Headers in C



```
#include <stdio.h>
#include <math.h>

int main()
{
    int number;

    printf("Enter an Integer\n");
    scanf("%d", number);
    printf("Square of %d is %d\n", number, sqrt(number));

    return 0;
}
```

Constants in C

Example program using const keyword in C:

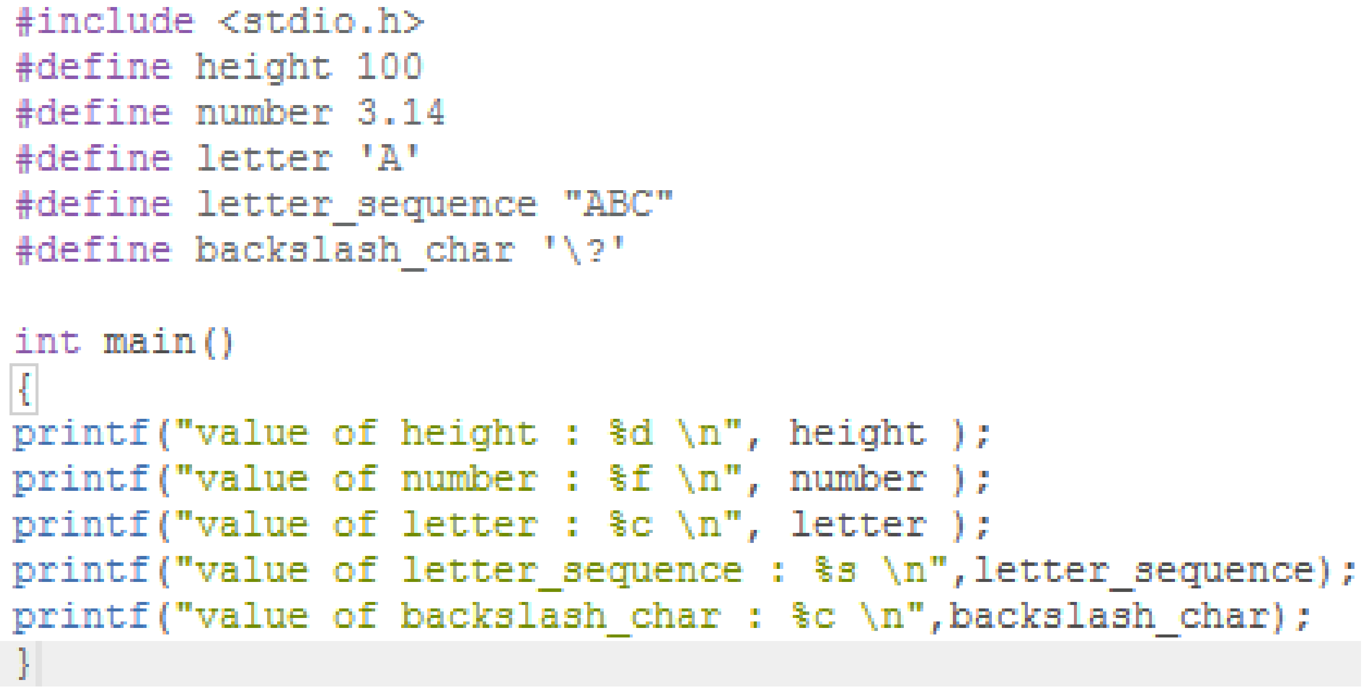


```c
#include <stdio.h>
int main()
{

const int height = 100; /*int constant*/
const float number = 3.14; /*Real constant*/
const char letter = 'A'; /*char constant*/
const char letter_sequence[10] = "ABC"; /*string constant*/
const char backslash_char = '\?'; /*special char cnst*/
printf("value of height :%d \n", height );
printf("value of number : %f \n", number );
printf("value of letter : %c \n", letter );
printf("value of letter_sequence : %s \n", letter_sequence);
printf("value of backslash_char : %c \n", backslash_char);

}
```

Output:

```
value of height :100
value of number : 3.140000
value of letter : A
value of letter_sequence :
ABC
value of backslash_char : ?
```

Example program using #define preprocessor directive in C:



```c
#include <stdio.h>
#define height 100
#define number 3.14
#define letter 'A'
#define letter_sequence "ABC"
#define backslash_char '\?'

int main()
{
printf("value of height : %d \n", height );
printf("value of number : %f \n", number );
printf("value of letter : %c \n", letter );
printf("value of letter_sequence : %s \n",letter_sequence);
printf("value of backslash_char : %c \n",backslash_char);
}
```

Output:

```
value of height :100
value of number : 3.140000
value of letter : A
value of letter_sequence :
ABC
value of backslash_char : ?
```