Computer Programming using C

Instructor: HOU, Fen

2025

Your First C Program

- Program hello.c

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

These is a comment. Text that is bracketed by the starting symbol pair /* and the ending symbol pair */ is ignored by the complier.

Every program has a function named main, where execution begins. The parentheses following main indicate to the complier that it is a function. The keyword int declares the return type to be integer. The key word void indicates the function takes no arguments

Start the body of the function

Invoke/call the function printf to print a string on the screen

End the body of the function

```
Hello, world!
```

Your First C Program

- Program hello.c

These is a comment. Text that is bracketed by the starting symbol pair /* and the ending symbol pair */ is ignored by the complier.

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

```
Hello, world!
```

Your First C Program

- Program hello.c

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

Lines that begin with a # are called preprocessing directives. They communicate with the preprocessor. Here is to cause the preprocessor to include a copy of the standard header file stdio.h in the code. The stdio.h is needed since it contains the information about the printf() function.

```
Hello, world!
```

Your First C Program

- Program hello.c

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

Every program has a function named main, where execution begins. The parentheses following main indicate to the complier that it is a function. The keyword int declares the return type to be integer. The key word void indicates the function takes no arguments

```
Hello, world!
```

Your First C Program

- Program hello.c

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

Start the body of the function

```
Hello, world!
```

Your First C Program

- Program hello.c

```
1   /* The traditional first program in honor of
2    Dennis Ritchie who invented C at Bell Labs
3   in 1972 */
4
5   #include <stdio.h>
6
7   int main(void)
8   {
9    printf("Hello, world!\n");
10   return(0);
11  }
```

Invoke/call the function printf to print a string on the screen

End the body of the function

```
Hello, world!
```

Exercise

- Write a program that displays the following sentence:

```
She sells sea shells by the seashore.
```

Exercise

- Exercise

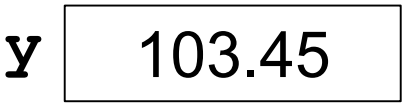
```
1   #include <stdio.h>
2
3   int main(void)
4   {
5    printf(" She sells sea shells by the seashore. \n");
6    return(0);
7    }
```

```
She sells sea shells by the seashore.
```

Variables, Expressions, and Assignments

- What is a variable?
  - Storage cells in the main memory
  - A variable can be viewed as a box for holding some values

  - Depending on the type of data it is storing, a variable can occupy one or more bytes



What is a Variable?

- Any sensible program maintains a number of variables.
- A program can be regarded as a "series" of program statements for modifying the values of variables.

Use Variables to Manipulate Integer Values

- Program ex3.c

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6   side = 3;
7    perimeter = 4 * side;
8    area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12    return(0);
13   }
```

```
Side : 3
Perimeter: 12
Area : 9
```

```
side = 3;
perimeter = 4 * side;
area = side * side;
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6    side = 3;
7    perimeter = 4 * side;
8    area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12   return(0);
13   }
```

Line 5: Variable declaration. Side. Perimeter, and area are called variables. A variable name consists of a sequence of letters, digits and underscores (_).

```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6      side = 3;
7      perimeter = 4 * side;
8      area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12    return(0);
13   }
```

Line 5: Variable declaration. The type of each variable must be specified. Here int is the integer data type.

```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6      side = 3;
7      perimeter = 4 * side;
8      area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12    return(0);
13   }
```

Line 5: Variable declaration. A variable name should be meaningful. Meaningless variable names make code difficult to read.

```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

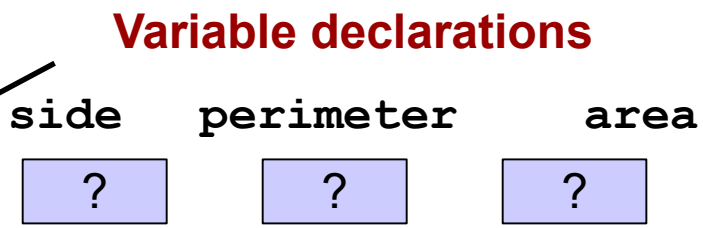
```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6    side = 3;
7    perimeter = 4 * side;
8    area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11   printf("Area : %d\n", area);
12   return(0);
13   }
```

Line 5: Variable declaration: Create a variable by giving it a name and specifying its type.
All declarations and statements in C end with a semicolon.

```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

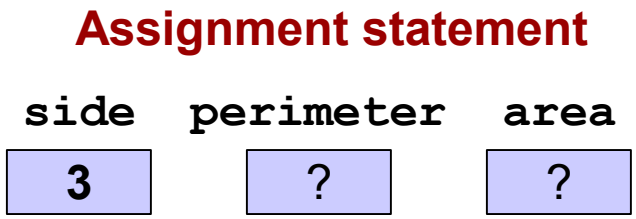
```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6    side = 3;
7    perimeter = 4 * side;
8    area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12   return(0);
13   }
```




```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6      side = 3;
7      perimeter = 4 * side;
8      area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12    return(0);
13   }
```

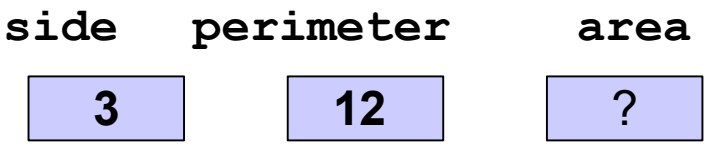



```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6      side = 3;
7      perimeter = 4 * side;
8      area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11   printf("Area : %d\n", area);
12   return(0);
13   }
```

Expression on the R.H.S. evaluates to 12.

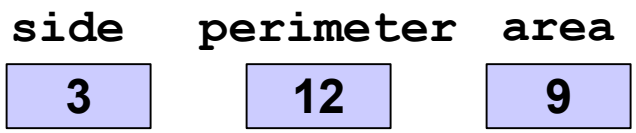



```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5   int side, perimeter, area;
6      side = 3;
7      perimeter = 4 * side;
8      area = side * side;
9     printf("Side : %d\n", side);
10    printf("Perimeter: %d\n", perimeter);
11    printf("Area : %d\n", area);
12    return(0);
13   }
```

R.H.S. expression evaluates to 9.




```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

- The printf() statement

```
printf("Side : %d\n", side);
```

- This printf() statement has two arguments which are separated by a comma.

```
printf("Side : %d\n", side);
```

  - The first argument is called the format string.
  - In this example, the format string contains the format specier %d, which species that the value of the corresponding expression is to be printed in the format of a decimal integer

```
Side : 3
Perimeter: 12
Area : 9
```

Use Variables to Manipulate Integer Values

```
printf("Side : %d\n", side);
```

- The second argument is the variable side. The expression whose value is to be supplied to the format string.

```
Side : 3
Perimeter: 12
Area : 9
```

The general form of a simple C program

```
1   # preprocessing directives
2
3   int main(void)
4   {
5    variable declarations
6    statement 1;
7    statement 2;
8    statement 3;
9    statement 4;
10   ...
11  return(0);
12  }
```

Exercise

- Write a program that displays the following sentences

```
She sells sea shells by the seashore.
```

Exercise

- Exercise

```
1   #include <stdio.h>
2
3   int main(void)
4   {
5    printf(" She sells sea shells by the seashore. \n");
6    return(0);
7    }
```

```
She sells sea shells by the seashore.
```