Jiangxi University of Science and Technology

Ch06 Modularity Using Functions

Lecture0604 Standard Library Functions_B

THE C PROGRAMMING LANGUAGE

6.4 Standard Library Functions

➢Scaling

—The method for adjusting the random numbers produced by a random-number generator to reside within a specified range is called ***scaling***

- To scale a random number as an integer value ***between 1 and N***:
  - **1 + (int)rand() % N**

- To produce a random integer ***between the numbers a and b***:
  - **a + (int)(rand() % (b - a + 1))**

6.4 Standard Library Functions

➢ Program 6.9 Coin Toss Simulation

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  #include <time.h>
4.  int flip(int);   /*prototype for flip*/
5.  void percentages(int, int);//prototype for percentage
6.  int main(){
7.          int numTosses = 100000;
8.          int heads;
9.          heads = flip(numTosses);
10.         percentages(numTosses, heads);
11.         return 0;
12. }
```

6.4 Standard Library Functions

```
13. int flip(int numTimes){
14.         int randValue;
15.         int heads = 0;
16.         srand(time(NULL));
17.         for (int i = 1; i <= numTimes; i++){
18.                 randValue =  (int)rand() % 2;
19.                 if (randValue >=1)
20.                         heads++;
21.         }
22.         return (heads);
23. }
```

this method tosses the coin numTimes and returns the number of heads

6.4 Standard Library Functions

Program 6.9 Coin Toss Simulation

```
24.  void percentages(int numTosses, int heads){
25.          int tails;
26.          float perheads, pertails;
27.          if (numTosses == 0)
28.                  printf("There were no tosses, so no percentages can be
                     calculated.\n");
29.          else
30.          {
31.                   tails = numTosses - heads;
32.                  printf("Number of coin tosses: %d\n", numTosses);
33.                  printf("   Heads: %d   Tails: %d\n", heads, tails);
34.                 perheads=(float)heads/numTosses *100.0;
35.                 pertails=(float)(numTosses-heads) /numTosses * 100.0;
36.                  printf("Heads came up %6.2f percent of the time.\n", perheads);
37.                 printf("Tails came up %6.2f percent of the time.\n", pertails);
38.          }
39.  }
```

6.4 Standard Library Functions

➢Input/Output Library Functions

- **getchar()** can be used for single character input
- **int** getchar()
- The reason for returning characters in integer format is to allow the End-Of-File (EOF) sentinel to be returned

- **putchar()** expects a single character argument and displays the character passed to it on the terminal
- For example: putchar('a')

6.4 Standard Library Functions

➢ Character Processing Functions (ctype.h)

Table 6.2 Character Functions (require the header file `ctype.h`)

Prototype	Description	Example
`int isalnum(int)`	Returns a non-0 number if the argument is a letter or a digit; otherwise it returns a 0.	`isalnum('9');`
`int isalpha(int)`	Returns a non-0 number if the argument is a letter; otherwise, it returns 0.	`isalpha('a')`
`int iscntrl(int)`	Returns a non-0 number if the argument is a control argument; otherwise, it returns 0.	`iscntrl('a')`
`int isdigit(int)`	Returns a non-0 number if the argument is a digit (0–9); otherwise, it returns 0.	`isdigit('a')`
`int isgraph(int)`	Returns a non-0 value if the argument is a printable character other than a space; otherwise it returns a 0.	`isgraph('@')`
`int islower(int)`	Returns a non-0 number if the argument is lowercase; otherwise, it returns 0.	`islower('a')`

6.4 Standard Library Functions

➢ Character Processing Functions

Table 6.2 Character Functions (require the header file `ctype.h`) (continued)

Prototype	Description	Example
`int isprint(int)`	Returns a non-0 number if the argument is a printable argument; otherwise, it returns 0.	`isprint('a')`
`int ispunct(int)`	Returns a non-0 number if the argument is a punctuation argument; otherwise, it returns 0.	`ispunct('!')`
`int isspace(int)`	Returns a non-0 number if the argument is a space; otherwise, it returns 0.	`isspace(' ')`
`int isupper(int)`	Returns a non-0 number if the argument is uppercase; otherwise, it returns 0.	`isupper('a')`
`int isxdigit(int)`	Returns a non-0 value if the argument is a hexadecimal digit (A–F, a–f, or 0–9).	`isxdigit('b')`
`int tolower(int)`	Returns the lowercase equivalent if the argument is uppercase; otherwise, it returns the argument unchanged.	`tolower('A')`
`int toupper(int)`	Returns the uppercase equivalent if the argument is lowercase; otherwise, it returns the argument unchanged.	`toupper('a')`

6.4 Standard Library Functions

➢Program 6.10 The character is a letter or digit

```
1.  #include <stdio.h>
2.  #include <ctype.h>
3.  int main()
4.  {
5.          char inChar;
6.          do
7.          {
8.                  printf("\nPush any key (type an x to stop) ");
9.                  inChar = getchar();
10.                 inChar = tolower(inChar);
11.                 getchar();
12.         if ( isalpha(inChar))
13.                 printf("\nThe character entered is a letter.\n");
14.         else if ( isdigit(inChar) )
15.                 printf("\nThe character entered is a digit.\n");
16.         }while (inChar != 'x');
17. }
```

/*get and ignore the ENTER key */

/*You can use **EOF** replace **'x'**/

6.4 Standard Library Functions

- Conversion Functions

Table 6.3 String Conversion Functions (require the header file `stdlib.h`)

Prototype[8]	Description	Example
`int atoi(string)`	Converts an ASCII string to an integer. Conversion stops at the first noninteger character.	`atoi("1234")`
`double atof(string)`	Converts an ASCII string to a double-precision number. Conversion stops at the first character that cannot be interpreted as a double.	`atof("12.34")`
`string itoa(int)`	Converts an integer to an ASCII string. The space allocated for the returned string must be large enough for the converted value.	`itoa(1234)`

6.4 Standard Library Functions

➢ Program 6.11 Conversion Functions

```
1.  #include <stdio.h>
2.  #include <stdlib.h>
3.  int main(){
4.  int num;  double dnum;
5.  num = atoi("1234");
6.  printf("The string \"1234\" as an integer number is: %d \n", num);
7.  printf("This number divided by 3 is: %d \n\n", num/3);
8.  dnum = atof("1234.96");
9.  printf("The string \"1234.96\" as a double is: %f \n", dnum);
10. printf("This number divided by 3 is: %f \n", dnum/3);
11. return 0;
12. }
```

6.5 Summary

- A function is called by giving its name and passing any data to it in the parentheses following the name
- The first line of the function is called the function header
- A function's return type is the data type of the value returned by the function
- Functions can directly return at most a single value to their calling functions
- Functions can be declared to all calling functions with a function prototype
- Arguments passed to a function provide a means of evaluating any valid C expression
- A set of preprogrammed functions for
  - mathematical calculations
  - character input and output
  - character processing
  - numerical conversions

Reference

- BOOK
- Some part of this PPT given by Prof 欧阳城添

(Prof: Chengtian Ouyang)

- with special thank
- https://www.codingunit.com/c-tutorial-first-c-program-hello-world