Input-Output

Structured Programming Language (CSE-1271)

Course Instructor : Mohammed Mamun Hossain
Assistant Professor, Dept. of CSE, BAUST

Outline

1. Data Type and Range
2. Basic Input and Output
3. Header Files

Data Type and Ranges

Integer Type:

Type	Storage size	Value range
char	1 byte	-128 to 127 or 0 to 255
unsigned char	1 byte	0 to 255
signed char	1 byte	-128 to 127
int	2 or 4 bytes	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
unsigned int	2 or 4 bytes	0 to 65,535 or 0 to 4,294,967,295
short	2 bytes	-32,768 to 32,767
unsigned short	2 bytes	0 to 65,535
long	4 bytes	-2,147,483,648 to 2,147,483,647
unsigned long	4 bytes	0 to 4,294,967,295

Data Type and Ranges

Float Type:

Type	Storage size	Value range	Precision
float	4 byte	1.2E-38 to 3.4E+38	6 decimal places
double	8 byte	2.3E-308 to 1.7E+308	15 decimal places
long double	10 byte	3.4E-4932 to 1.1E+4932	19 decimal places

The % Format Specifiers

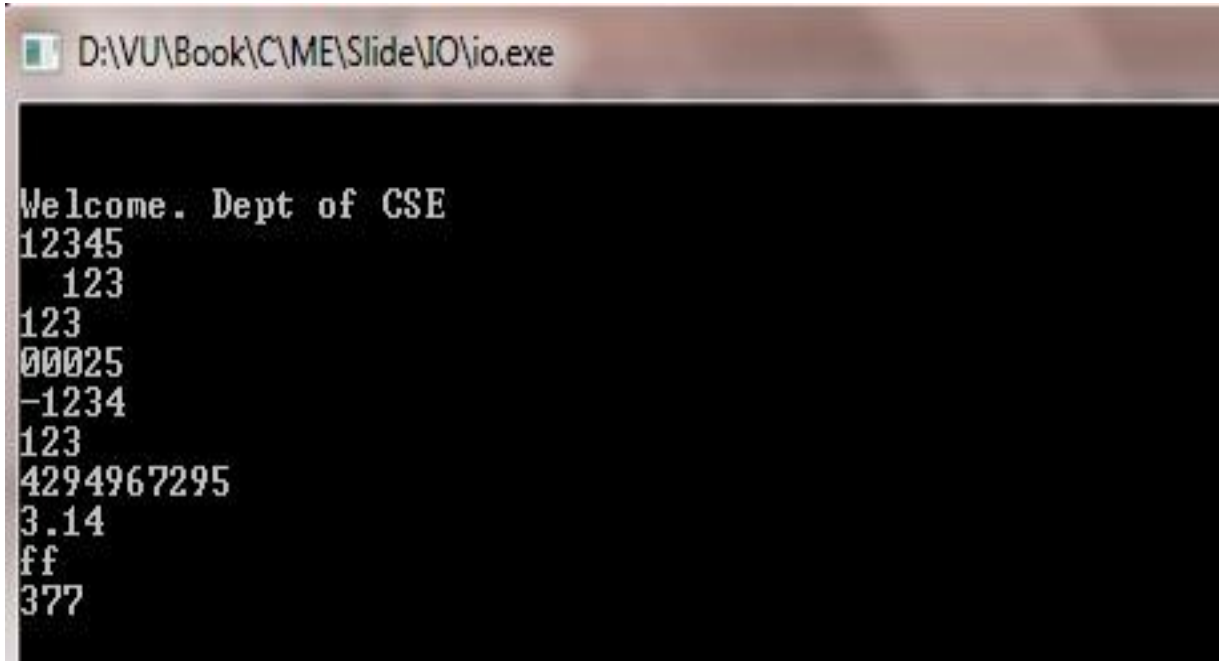
- %c char single character
- %d (%i) int signed integer
- %e (%E) float or double exponential format
- %f float or double signed decimal
- %g (%G) float or double use %f or %e as required
- %o int unsigned octal value
- %p pointer address stored in pointer
- %s array of char sequence of characters
- %u int unsigned decimal
- %x (%X) int unsigned hex value

The % Format Specifiers

- %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)

Format Specifiers Example

```c
#include<stdio.h>

int main()
{
    printf("\n\n");
    printf("%s\n", "Welcome. Dept of CSE");
    printf("%d\n", 12345);

    printf("%5d\n", 123);
    printf("%-5d\n", 123);

    printf("%05d\n", 25);

    printf("%i\n", -1234);
    printf("%u\n", 123);
    printf("%u\n", -1);

    printf("%3.2f\n", 3.14159);
    printf("%x\n", 255);
    printf("%o\n", 255);

    return 0;
}
```



Library Function with Header file

- **stdio.h: I/O functions:**
  - ✓ **getchar()** returns the next character typed on the keyboard.
  - ✓ **putchar()** outputs a single character to the screen.
  - ✓ **printf()**
  - ✓ **scanf()**
- **string.h: String functions**
  - ✓ **strcat()** concatenates a copy of str2 to str1
  - ✓ **strcmp()** compares two strings
  - ✓ **strcpy()** copys contents of str2 to str1

Library Function with Header file

- **ctype.h: Character functions**
  - ✓ **isdigit()** returns non-0 if arg is digit 0 to 9
  - ✓ **isalpha()** returns non-0 if arg is a letter of the alphabet
  - ✓ **isalnum()** returns non-0 if arg is a letter or digit
  - ✓ **islower()** returns non-0 if arg is lowercase letter
  - ✓ **isupper()** returns non-0 if arg is uppercase letter
- **math.h: Mathematics functions**
  - **cos()** returns cosine of arg
  - **exp()** returns natural logarithm e
  - **fabs()** returns absolute value of num
  - **sqrt()** returns square root of num
  - **pow**() returns power of num as specified

Library Function with Header file

- **time.h: Time and Date functions**
  - ✓ **time()** returns current calendar time of system
  - ✓ **difftime()** returns difference in secs between two times
  - ✓ **clock()** returns number of system clock cycles since program execution
- **stdlib.h: Miscellaneous functions**
  - ✓ **malloc()** provides dynamic memory allocation
  - ✓ **rand()** generates random numbers
  - ✓ **srand()** used to set the starting point for rand()

The scanf() & printf() functions

- The C library function scanf() reads input from stdin.
- The C library function printf() sends output to stdout..

```c
#include <stdio.h>

int main ()
{
   int number, n1, n2;

   scanf("%d",&number);
   printf("number = %d\n\n", number);

   scanf("%d%d",&n1, &n2);
   printf("n1 = %d   n2 = %d\n\n",n1, n2);

   printf("number = %d   n1 = %d   n2 = %d\n\n",number, n1, n2);

   scanf("%d%d%d",&n1, &n2, &number);
   printf("n1 = %d   n2 = %d   number = %d\n",n1, n2, number);
   printf("n1 = %d   n2 = %d   number = %d\n",n1, number, n2);


   return 0;
}
```

```
45
number = 45

100 200
n1 = 100   n2 = 200

number = 45   n1 = 100   n2 = 200

20 -80 500
n1 = 20   n2 = -80   number = 500
n1 = 20   n2 = 500   number = -80
```

The scanf() & printf() functions

```c
#include <stdio.h>

int main ()
{
    int a, b;

    printf("Enter the value of a & b (two integer) : ");
    scanf("%d%d",&a,&b);

    printf("%d %d\n\n", a,b);

    a=a+b;
    printf("%d %d\n\n", a,b);

    printf("%d %d\n\n", ++a,b++);

    printf("%d %d\n\n", a,b);

    return 0;
}
```

```
Enter the value of a & b (two integer) : 50 100
50 100

150 100

151 100

151 101
```

The getchar() & putchar() functions

- The getchar() function reads only single character at a time.
- The putchar(c) function displays single character on the screen.

Programming example with getchar( ) and putchar()

```
char c;
printf( "Enter a value :");
c = getchar();
printf( "\nYou entered: ");
putchar(c);
```

The gets() & puts() functions

- The gets() function reads a line of text .
- The puts() function displays a line of text.

```
char str[100];
 printf( "Enter a value :");
 gets(str);
 puts(str);
```

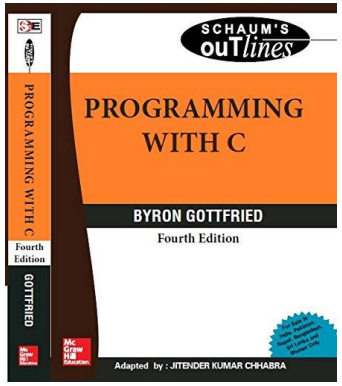
Thank You.

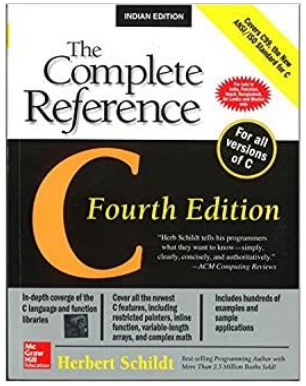
Questions and Answer

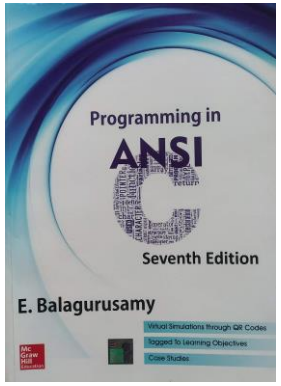
References

Books:

1. Programming With C. *By Byron Gottfried*
2. The Complete Reference C. *By Herbert Shield*
3. Programming in ANSI C *By E. Balagurusamy*
4. Teach yourself C. *By Herbert Shield*







Web:

1. www.wikbooks.org

and other slide, books and web search.