To:
Tenouk

# C LAB WORKSHEET 6_2
# C scanf(), scanf_s() Exercises 3

Items in this page:

1. More exercises and activities on scanf() and scanf_s() functions.
2. scanf(), scanf_s() and strings and characters issues.
3. #define preprocessor directives.
4. Tutorial references are: C/C++ intro & brief history, C/C++ data type 1, C/C++ data type 2, C/C++ data type 3 and C/C++ statement, expression & operator 1, C/C++ statement, expression & operator 2 and C/C++ statement, expression & operator 2. More scanf() and its family examples can be found in C formatted input/output.

17. In a scanf(), use %f instead of %.2f when reading in a float. Also don't use the '\n' character. Try the following code.

```
#include <stdio.h>

int main(void)
{
    int i; float f;
    printf("Please enter a whole number: ");
    scanf("%d", &i);
    printf("Please enter a number ");
    printf("with a decimal point: ");
    scanf("%f", &f);
    printf("The sum of both entered number
is = %f\n", i+f);
    return 0;
}
```

```
Please enter a whole number: 1234
Please enter a number with a decimal point: 12.34
The sum of both entered number is = 1246.340000
Press any key to continue . . .
```

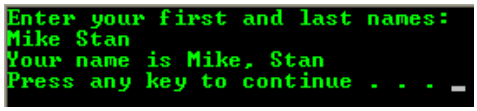
18. Try the following code. When scanning in strings or character, add a space before the format specifier. Don't use "%s", but use " %s" (note the space). The char x[12] is an array variable, where we can store up to 11 character continuously. Another one character is reserved for null, '\0' that used to terminate a string.

```
#include <stdio.h>

int main(void)
{
    char x[12];
    printf("Enter your name with no spaces:
");
    scanf(" %s", x);
    printf("Your name is %s.\n", x);
    return 0;
}
```

```
Enter your name with no spaces: MikeStan
Your name is MikeStan.
Press any key to continue . . .
```

19. When running the following program, enter your first and last name separated by space.

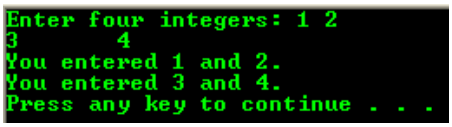
```
#include <stdio.h>

int main(void)
{
    char MyFname[20];
    char MyLname[20];
    printf("Enter your first and last names:
\n");
    scanf(" %s %s", MyFname, MyLname);
    printf("Your name is %s, %s\n",
MyFname, MyLname);
    return 0;
}
```

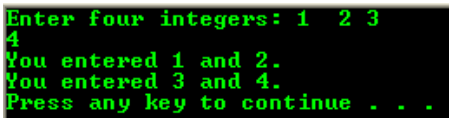



20. Run the following code two times and enter the numbers as shown with spaces and tabs. For the first run, for example, type these characters in order:

    1(press space)2(press enter)3(press tab)4 (press enter)

```
#include <stdio.h>

int main(void)
{
    int i, j, k, l;
    printf("Enter four integers: ");
    scanf("%d%d", &i, &j);
    scanf("%d%d", &k, &l);
    printf("You entered %d and %d.\n", i, j);
    printf("You entered %d and %d.\n", k, l);
    return 0;
}
```



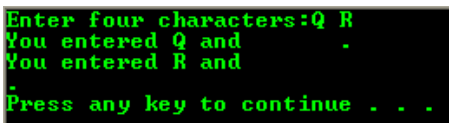




    a. What were the values of i, j, k and l during the first run?
    b. What were the values of i, j, k and l during the second run?
    c. While scanf() is looking for an integer, does it pass over spaces? Or over tabs? Or over carriage return?

a. i = 1, j = 2, k = 3 and l = 4.
b. Same as in the first run.
c. It passes over spaces, tabs and carriage return.

21. Run the following program and enter Q (press tab)R(press enter)S(press space)T.

```
#include <stdio.h>

int main(void)
{
    char a, b;
    char c, d;
    printf("Enter four characters:");
    scanf("%c%c", &a, &b);
    scanf("%c%c", &c, &d);
    printf("You entered %c and %c.\n", a, b);
    printf("You entered %c and %c.\n", c, d);
    return 0;
}
```




    a. What were the values of a, b, c, and d after the run?

a. a = Q, b = R then the program terminate.

22. Now force the scanf() to pass over whitespace by preceding the %c with a space (“ %c”).

```
#include <stdio.h>

int main(void)
{
    char a, b;
    char c, d;
    printf("Enter four characters:");
    scanf(" %c %c", &a, &b);
    scanf(" %c %c", &c, &d);
    printf("You entered %c and %c.\n", a, b);
    printf("You entered %c and %c.\n", c, d);
    return 0;
}
```

```
Enter four characters:Q R
S T
You entered Q and R.
You entered S and T.
Press any key to continue . . .
```

a. If we want to skip over whitespace, that is spaces, tabs and carriage returns and we are scanning for integers, then is it necessary to precede the %d by a space?
b. What about when we want to read in characters? Should we precede the %c with a space?

a. No.
b. Yes.

23. Run the following program and precede the %c’s with spaces.

```
#include <stdio.h>

int main(void)
{
    char a, b;
    int i, j;
    printf("Enter two char-int pairs: ");
    scanf(" %c %d", &a, &i);
    scanf(" %c %d", &b, &j);
    printf("%c:%d:\n", a, i);
    printf("%c:%d:\n", b, j);
    return 0;
}
```

```
Enter two char-int pairs: R8 T3
R:8:
T:3:
Press any key to continue . . .
```

a. Did the values get read into the variables as they should have been?
b. Try the same experiment again without the leading spaces in the format strings for integers e. g. scanf(" %c%d", &a, &i);). Did you get the results as before?
c. Try the same experiment again without the leading spaces in the format strings for the characters (e.g. scanf("%c %d", &a, &i);). Did you get the same result as before?
d. When reading in integers, spaces are not needed, true or false?
e. When reading in characters, we would add the spaces before the %c’s, true or false? Format strings for floats behave like integers and those for strings behave like characters.

a. Yes.
b. Yes

```
Enter two char-int pairs: R8 T3
R:8:
T:3:
Press any key to continue . . .
```

c. No.

```
Enter two char-int pairs: R8 T3
R:8:
 :-858993460:
Press any key to continue . . .
```

d. True.
e. True.

24. Run the following code, and then re-run it by removing one & (at-address operator) at a time.

```
#include <stdio.h>

int main(void)
{
    int i;
    char t[12];
    printf("Enter an integer and a string separated by space:");
    scanf("%d %s", &i, &t);
    printf("i was %d and t was %s: \n", i, t);
    return 0;
}
```

a. Do we need the & signs for scanning in integers? What about for strings?
b. From the (a) answer, do you think the methods of storing strings and integers are different from each other?

```
Enter an integer and a string separated by space:20 Test
i was 20 and t was Test:
Press any key to continue . . .
```

When the & removed from &i, the following Debug error displayed.


When the & removed from &t, the program run smoothly.

```
Enter an integer and a string separated by space:88 String
i was 88 and t was String:
Press any key to continue . . .
```

a. We need it for integers and not for strings.
b. Yes. A string declared as a char array. An **array name** is a pointer to the first element of the array. Characters stored contiguously for strings

25. Try the following code that contain #define preprocessor directive.

```
#include <stdio.h>
#define PI 3.14
#define RADIUS "radius"

int main(void)
{
    float flt;
    printf("Enter the radius: ");
    scanf("%f", &flt);
    printf("The %s is %.2f\n", RADIUS, flt);
    printf("The circumference is: %.2f\n", 2*flt*PI);
    printf("The area is: %.2f\n", PI*flt*flt);
    return 0;
}
```

a. If we wanted to add more digits after the decimal point for 3.14 for better precision, in how many places would we have to make that change?
b. Do we have to specify the data type with #define directives?
c. Change the radius of the preprocessor directive to other string and rebuild your program. What is the changes?
d. Change the value of 3.14 of the preprocessor directive to other value and rebuild your program. What is the changes?
e. Does your compiler allow you to change the value of PI by adding the following statement inside main()?

PI=3.1416

```
Enter the radius: 4
The radius is 4.00
The circumference is: 25.12
The area is: 50.24
Press any key to continue . . .
```

a. One place only that is in the #define PI 3.14.
b. No.
c. We change the #define RADIUS "radius" to #define RADIUS "newradius"

and rebuild the program. The output is same.

d. We change #define PI 3.14 to #define PI 3.1427. The output is shown below. All the PI's value in the program has been changed to 3.1427.

```
Enter the radius: 4
The newradius is 4.00
The circumference is: 25.14
The area is: 50.28
Press any key to continue . . .
```

f. No, because we re-define the PI constant.

26. Study the following directive (macros).

```
#include <stdio.h>
#define SQUARE(x) x*x
#define CUBE(x) x*x*x
#define FOURTH(x) x*x*x*x
#define FORMAT_I "i is equal to %d\n"
#define PRINT_I printf(FORMAT_I, i); //
also called a macro

int main()
{
    int i = 5;
    printf(FORMAT_I, i);
    i = CUBE(i);
    printf(FORMAT_I, i);
    i = FOURTH(5);
    printf(FORMAT_I, i);
    return 0;
}
```

For every occurrence of FORMAT_I, the string “i is equal to %d\n” is substituted and in place of CUBE(i), i*i*i is substituted, since i is used in place of x in the #define CUBE (x) x*x*x directive. Similar to FOURTH(5). The previous code can be expanded as shown below.

```
#include <stdio.h>
#define SQUARE(x) x*x
#define CUBE(x) x*x*x
#define FOURTH(x) x*x*x*x
#define FORMAT_I "i is equal to %d\n"
#define PRINT_I printf(FORMAT_I, i); //
also called a macro

int main()
{
    int i = 5;
    printf("i is equal to %d\n", i);
    i = i*i*i;
    printf("i is equal to %d\n", i);
    i = 5*5*5*5;
    printf("i is equal to %d\n", i);
    return 0;
}
```

```
i is equal to 5
i is equal to 125
i is equal to 625
Press any key to continue . . .
```

```
i is equal to 5
i is equal to 125
i is equal to 625
Press any key to continue . . .
```

We can see that both programs have similar output.

27. Some questions for conclusion.

a. What is a variable’s content called?
b. What is the location where a variable is stored in memory called?
c. What is the operator used to see this location?
d. By what is variable called?

a. A value.
b. An address, a memory address.
c. The address-of operator (&).
d. By its name.

e. Define a directive called TAX_RATE and set it to 0.05. Define a string variable called part[ ], a float called price and an integer called quantity. Total sale is equal to quantity times price, plus the tax on the total sale. On the firs line of the screen ask the user to input the name of the part, on the second line ask for the price of the part and on the third line ask for the number of parts that were sold. Then, print a report that has the following format.

```
Input the part name: buttons
Input price per part: 0.50
How many parts were sold? 5

        5 buttons were sold for 1.50
        each.
        The total sales = 2.625, including
        sale tax.
```

```
#include <stdio.h>
#define TAX_RATE 0.05
int main()
{
   // dummy initial values
   char part[20] = "test";
   float price = 2.0;
   int quantity = 4;
   printf("Input the part name:");
   scanf_s("%19s", &part, 20);
   printf("Input price per part: ");
   scanf_s("%f", &price, sizeof(float));
   printf("How many parts were sold? ");
   scanf_s("%d", &quantity, 1);
   printf("\n\t%d %s were sold for %.2f each\n", quantity, part, price);
   printf("\tThe total sales = %.2f, including sale tax.\n", (quantity*price)+
                                        (TAX_RATE*quantity*price));
   return 0;
}
```

```
Input the part name:lamp
Input price per part: 20.25
How many parts were sold? 12

        12 lamp were sold for 20.25 each.
        The total sales = 255.15, including sale tax.
Press any key to continue . . .
```

f. Declare two strings called employee[ ] and company[ ] along with a float called rate and integer called hours. Initialize employee[ ] to Bob Dylan and assign Tenouk Enterprise to company[ ]. Ask for the employee rate and hours. Calculate his pay: as a 10% less than the product of rate and hours. Display the report as shown below.

```
Input Bob Dylan's rate and worked
hours: 7.00 10
Bob Dylan worked 10 hours
The rate of pay was USD7.00 per hour
The gross pay was USD70.00
The net pay was USD63.00
Bob Dylan work for Tenouk Enterprise.
```

```
#include <stdio.h>

int main()
{
   // dummy initial values
   char employee[20] = "Bob Dylan";
   char company[20] = "Tenouk Enterprise";
   float rate = 0.0;
   int hours = 0;
   printf("Input Bob Dylan's rate and worked hours: ");
   scanf_s("%f %d", &rate, &hours);
   printf("Bob Dylan worked %d hours\n", hours);
   printf("The rate of pay was USD%.2f per hour\n", rate);
   printf("The gross pay was USD%.2f\n", rate*hours);
   printf("The net pay was USD%.2f\n", (rate*hours) - (0.1*rate*hours));
   printf("%s work for %s.\n", employee, company);
   return 0;
}
```

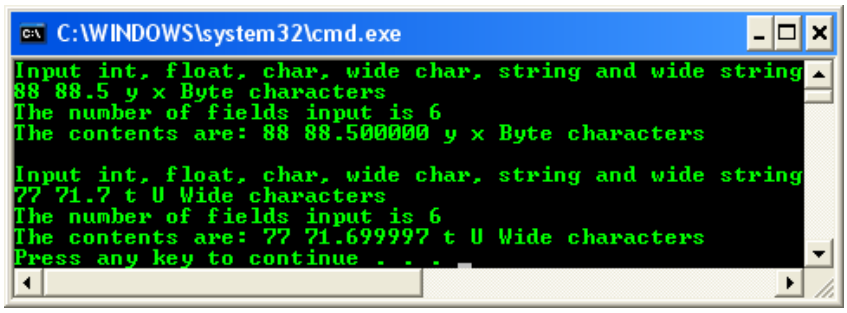
```
Input Bob Dylan's rate and worked hours: 7.00 10
Bob Dylan worked 10 hours
The rate of pay was USD7.00 per hour
The gross pay was USD70.00
The net pay was USD63.00
Bob Dylan work for Tenouk Enterprise.
Press any key to continue . . .
```

g. Study, build and run the following example and the output. Try finding the code that you don't understand and discuss with your partner/group member to find the answers. Then, create three questions with answers.

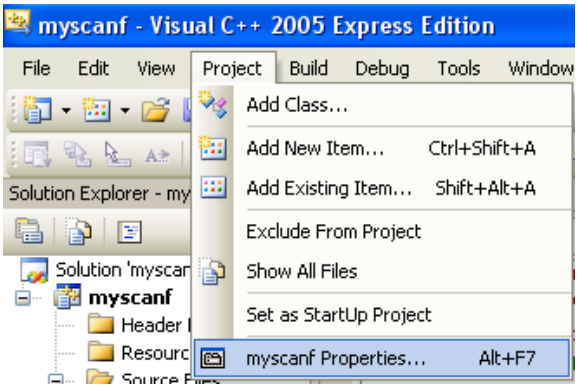
```
// scanf_s() and wscanf_s() functions to
read formatted input.
#include <stdio.h>

int main( void )
{
   int      i, result;
   float    fp;
   char     c, s[81];
   wchar_t  wc, ws[81];
   printf("Input int, float, char, wide char,
```

---

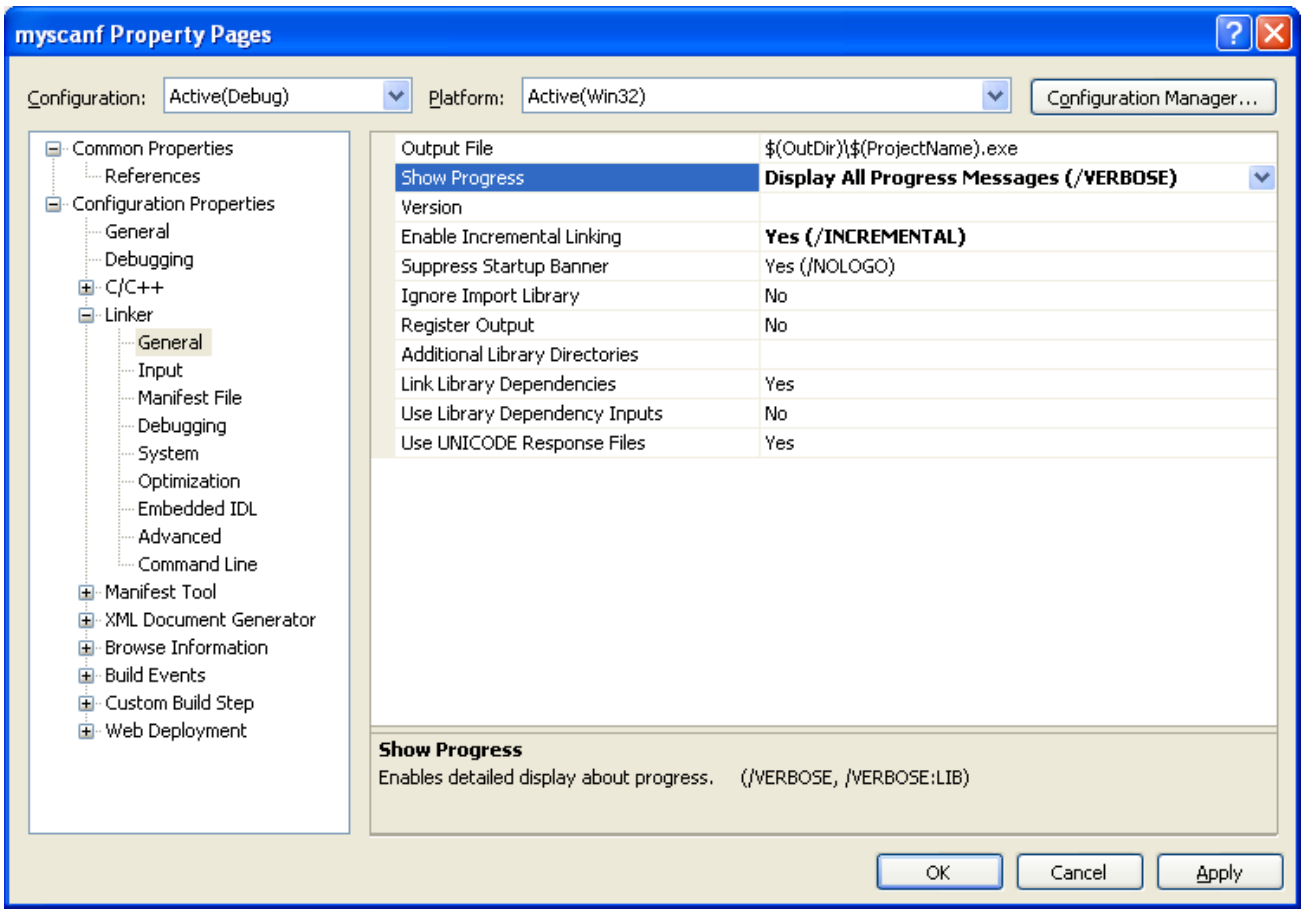
```
string and wide string\n");
   result = scanf_s("%d %f %c %C %s %
S", &i, &fp, &c, 1,
                  &wc, 1, s, 80, ws, 80);
   printf("The number of fields input is %d
\n", result);
   printf("The contents are: %d %f %c %
C %s %S\n", i, fp, c, wc, s, ws);
   printf("\nInput int, float, char, wide
char, string and wide string\n");
   result = wscanf_s(L"%d %f %hc %lc %
S %ls", &i, &fp, &c, 2,
                   &wc, 1, s, 80, ws, 80);
   wprintf(L"The number of fields input is
%d\n", result);
   wprintf(L"The contents are: %d %f %C
%c %hs %s\n", i, fp, c, wc, s, ws);
   return 0;
}

// Try the following inputs:
//       88 88.5 y x Byte characters
//       77 71.7 t U Wide characters
```

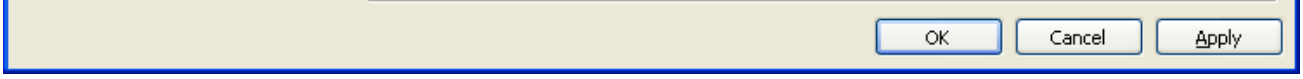



During your C/C++ building process you can see all the compiler activities. Select **Project** ⟶ ***your_project_name*** **Properties** menu.

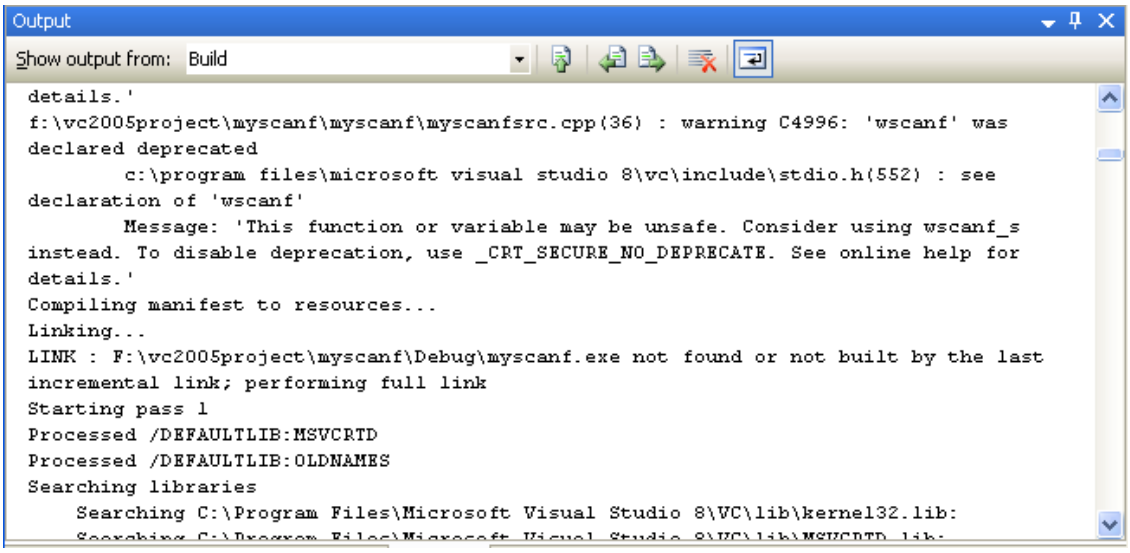



Expand the **Linker** folder under **Configuration Properties** folder. Select **General** link and on the right window, select **Display All Progress Messages (/VERBOSE)** of the **Show Progress**. Then click **Apply** button. Close the **Project Property Pages**.






Rebuild your program and see progress messages in the **Output** Window.




**| Main |< scanf() and scanf_s() 2 | A C/C++ Repetition: The for Loop >| Site Index | Download |**

**The C scanf() and scanf_s() family: Part 1 | Part 2 | Part 3**