

IBCS

File Processing

Mr. Brennan

Introduce WHILE loops and file processing

Part I

While Loops and Output Files

Description:

The WHILE..WEND loop
LINE INPUT
OPEN file\$ FOR OUTPUT AS #<channel>
PRINT #, output-to-write

Deliverable:

copyme.bas
copyme.out

Part II

Input Files

Description:

OPEN file\$ FOR INPUT AS #<channel>
EOF

Deliverable:

copyfile.bas

Part III

File Processing

Description:

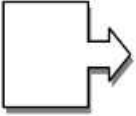


Command Line Arguments
INSTR
CLOSE

Deliverable:

scanfile.bas

Lab 5 File Processing

Instruction Conventions

| | |
|--|--|
|  | <p>When the “process” icon appears in the box on the left, then this box will contain one or more instructions that you will need to follow.</p> <p>The instructions will be as specific as is practical but could be different for different users and computer configurations.</p> |
|  | <p>When the “note” icon appears in the box to the left, this box will contain notes, hints, or tips that may be helpful to the lab activities.</p> |
|  | <p><user input></p> |
| | <p>When the keyboard icon appears in the box to the left, the box above will contain a line of input to be entered by the user, and this box will contain an explanation of what the user input will do.</p> |

IBCS

Lab 5 While Loops and Output Files

Part I



A WHILE loop repeats a group of statement as long as a condition is true. While loops are similar to FOR loops in that they are used to repeat a group of statements. The WEND statement indicates the end of the statements to be repeated. Unlike a FOR loop that will increment or decrement a loop variable, the WEND statement does not perform any action other than returning control of the program to the top of the WHILE loop.

Example: Read input from the user, and echo everything the user enters back to them. Stop the program when the user enters the phrase "stop program" but do not echo "stop program" to the user.

```
200 Print "Enter text to be echoed, or stop program to end the program"
300 input A$
400 while A$ <> "stop program"
500   print A$
600   input A$      : rem get the next line of user input
700   wend
```

IBCS

Lab 5 While Loops and Output Files

Part I



Decoding the echo.bas program:

Tell the user what to do:

```
200 Print "Enter text to be echoed, or stop program to end the program"
```

Read in the first line from the user:

```
300 input A$
```

Loop until the user enters "stop program":

<> means not equal
so as long as what the
user typed in does not
equal "stop program"
the loop will repeat:

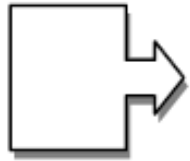
```
400 while A$ <> "stop program"
500   print A$
600   input A$      : rem  get the next line of user input
700   wend
```

Terminates the group of statements:
500: print out the value of A\$
which is what the user entered.
600: Gets the next line of input from
the user.
700: returns program control to line 400.

IBCS

Lab 5 While Loops and Output Files

Part I



1. Using a web browser, goto HWMath.net/IBCS and locate the file `echo.bas`

Save Link As `copyme.bas`

[Alternatively, you can open the `echo.bas` file, copy it into notepad, and save it to your program directory that has `blassic.exe`]



2. Open a command prompt window, change into your program directory, and issue the command:

```
C:\>blassic copyme.bas
```

- a. Run the program with a few input lines to verify that the program has been saved correctly and can be executed.
- b. Enter stop program to stop the program to verify that the program properly terminates when it should.
- c. Run the program with the following lines to be echoed:

This is a test

This is another test, longer than the first one

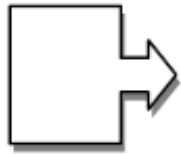
What if I enter a list, such as apples, pears, grapes

stop program

IBCS

Lab 5 While Loops and Output Files

Part I



3. Save this text as file echo.dat
This is a test
This is another test, longer than the first one
What if I enter a list, such as apples, pears, grapes
stop program



4. Run the copyme program again, but redirect the input to use the file echo.dat
`C:\>blassic copyme.bas < echo.dat`



Your output should be similar to the following:

```
Enter text to be echoed, or stop program to end the progr  
? This is a test  
? This is another test  
? What if I enter a list  
? Program terminating
```

Notice that not all parts of each line are echoed. the parts in red are missing, because the INPUT command stops at commas.

This is a test

This is another test, longer than the first one

What if I enter a list, such as apples, pears, grapes

stop program

IBCS

Lab 5 While Loops and Output Files

Part I



To read an entire line from the user, and not just up until the first comma, use the `LINE INPUT` command.



5. Change both lines 300 and 600 to use the `LINE INPUT` command:

```
LINE INPUT A$
```

6. Test your changes using your test data:

```
C:\>classic copyme.bas < echo.dat
```

Your output should be similar to the following:

```
Enter text to be echoed, or stop program to end the program
? This is a test
? This is another test, longer than the first one
? What if I enter a list, such as apples, pears, grapes
? Program terminating
```

If you have output similar to what is shown above then you are doing well and ready to move forward. 😊

IBCS

Lab 5 While Loops and Output Files

Part I



7. Run your program again, and rather than having the output go to the screen, redirect the output to file echo.out:

```
C:\>blassic copyme.bas < echo.dat > echo.out
```

Notice that no output went to the screen.

8. View the contents of the echo.out file:
type echo.out

```
Enter text to be echoed, or stop program to end the program
? This is a test
? This is another test, longer than the first one
? What if I enter a list, such as apples, pears, grapes
? Program terminating
```



Notice that the output file contains the ? that would have been sent to the screen as well as the echoed A\$ contents. This is because ALL data that would have been sent to the screen was redirected to the file echo.out.

IBCS

Lab 5 While Loops and Output Files

Part I



You are going to change your `copyme.bas` file to open a file within your program, rather than using the operating system to redirect the output. This means that the `?` to prompt the user for input will go to the screen, while only the echoed data will be written to the file.

In BASIC, we can associate a file number with a file name, so rather than telling the program the name of the file to write to on every PRINT statement, we can use a number. This file number (which is really called a **channel**) is associate with a file name on the `OPEN` command.

IBCS

Lab 5 While Loops and Output Files

Part I



The syntax of the open command is:

```
OPEN file$ FOR mode AS #channel
```

where

file\$ is a string containing the name of the file

mode is how we are going to open the file, for this

next step we will open the file for OUTPUT

which will overwrite the output file each time

the program is run.

#Channel is the file number we will use. For the

output file use channel 2.



9. Add line 190 to copyme.bas to open a file for output:

```
190 OPEN "copyme.out" FOR OUTPUT AS #2
```

IBCS

Lab 5 While Loops and Output Files

Part I



10. Change line 500 of copyme.bas to print to the output file:

```
500 print #2, A$
```



Note: we are only changing line 500, we still want the output from line 200 to go to the "standard output"

```
200 Print "Enter text to be echoed, or stop program to end the program"
```



11. Run the program without redirecting either the input or the output:

```
c:\>blassic copyme.bas
```

Example:

```
Enter text to be echoed, or stop program to end the program
? This is line 1
? this is line 2
? stop program
Program terminating
```

IBCS

Lab 5 While Loops and Output Files

Part I



12. Verify that the program created file `copyme.out`
`c:\>dir copyme.out`

Verify that the contents of `copyme.out`
`c:\>type copyme.out`

Sample Output:

```
This is line 1  
this is line 2
```



13. Run the program with the same input file
`c:\>blassic copyme.bas < echo.dat`

Repeat step 12 to verify the program's execution

14. Submit `copyme.bas` and `copyme.out` for grading.

Attachments

echo.bas