Designs 3 (Looping)

Graded course work

Important Notes:

- **Write all the programs, from the designs in this handout**
- **Use the same program names, and variable names that I have specified**
- **Compile, run, and test your programs**
- **Submit a copy of these programs to me for grading**

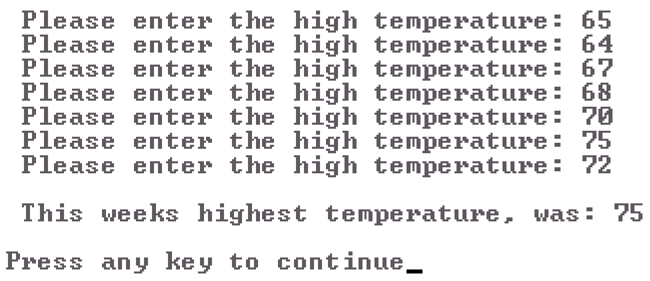
If you don’t know how to do any particular part, ask me -- I will show you how

Program (`highest2.c`)

Write a program that asks for 7 day time high temperatures, and prints out the weekly high temperature. Use a `for()` loop in this program.

```
declare floats "high_temperature", "highest_high_temperature"
put lower than possible value (temperature) into the box "highest_high_temperature"    -100
declare integer "loop_counter"

for(loop_counter 1 to 7)
        ask for today's highest temperature
        put the number in the box "high_temperature"

        if ("high_temperature" is greater than "highest_high_temperature")
                put "high_temperature" into the box "highest_high_temperature"
end for() loop

tell the number in box "highest_high_temperature"
```

Program output:



```
 Please enter the high temperature: 65
 Please enter the high temperature: 64
 Please enter the high temperature: 67
 Please enter the high temperature: 68
 Please enter the high temperature: 70
 Please enter the high temperature: 75
 Please enter the high temperature: 72

 This weeks highest temperature, was: 75

Press any key to continue_
```

Program (`price1.c`)

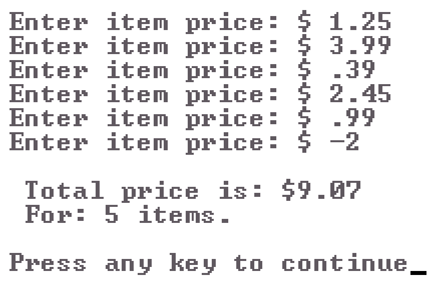
Write a program that will ask, and re-ask, the user to enter a price for an item of stock.

The program will keep track of the number of items, and the total price of the items.

When a negative number is entered for the price, the program will print the number of items, and the total price.

```
declare integer "number_of_items"
declare floats "item_price", "total_price"

ask for price of item
put value into the box "item_price"

while(while "item_price" is greater than zero)
        increment "number_of_items"
        add: "item_price" to "total_price"

        ask for price of item
        put value into the box "item_price"
end while() loop

tell the number "total_price"
tell the number "number_of_items"
```

Program output:



```
Enter item price: $ 1.25
Enter item price: $ 3.99
Enter item price: $ .39
Enter item price: $ 2.45
Enter item price: $ .99
Enter item price: $ -2

 Total price is: $9.07
 For: 5 items.

Press any key to continue_
```

Program (`validate2.c`)

Change program `validate1.c` so that it will re-ask the user for a month value if a value other than 1 to 12 is entered. Use a do - while() loop

```
declare integers "the_month", "flag" put 0 into box "flag"          0

do
        ask for a number for the month
        put the number in the box "the_month"

        if ("the_month" is between 1 and 12)
                tell "Ok -- good value"
                put 1 in box "flag"
        else
                tell "Not ok -- bad value"
while ("flag" equals zero)
```

Program (`numbers2.c`)

Extend the program `numbers1.c` so that it can handle numbers in the range 0 - 9.

Then add a `do - while()` loop to the program.

The program should stop when a negative number is entered at the prompt.

Program (`times3.c`)

Make a copy of program `times2.c`, and change it to use a `do - while()` loop.

Program (`menu1.c`)

Write a program that demonstrates the use of a menu in a program. A menu is just the text output of many printf() statements.

Step 1:

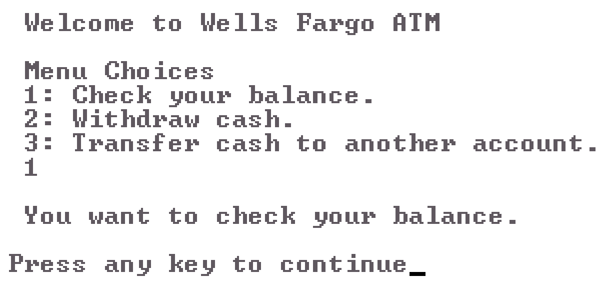
Just show the menu, and report the chosen option once.

Step 2:

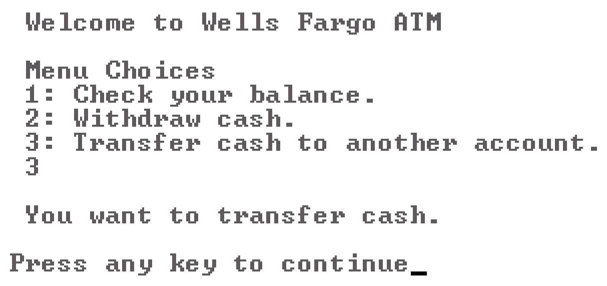
Then add a `do - while()` loop to the program, so that the menu will be re-shown after each choice.

The program should exit when a negative number is entered at the prompt.

Program output:



```
 Welcome to Wells Fargo ATM

 Menu Choices
 1: Check your balance.
 2: Withdraw cash.
 3: Transfer cash to another account.
 1

 You want to check your balance.

Press any key to continue_
```



```
 Welcome to Wells Fargo ATM

 Menu Choices
 1: Check your balance.
 2: Withdraw cash.
 3: Transfer cash to another account.
 3

 You want to transfer cash.

Press any key to continue_
```

Program (`for1.c`)

Step 1:

Based on program `total4.c`, write a program that prints out the value of its loop counter (1 - 5), as a vertical list of numbers.

Remember that loop counters must be an integers.

Step 2:

Change the program to print out the value of its loop counter (33 - 47), as a vertical list of numbers.

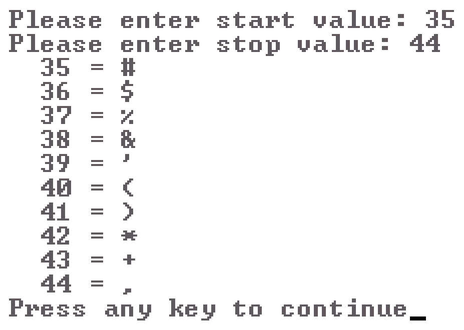
Step 3:

Change the program to print out the value of its loop counter (33 - 47), as a vertical list of characters.

Step 4:

Change the program, so that it asks the user for a *start value*, and a *stop value*, then prints out the value of its loop counter as a vertical list of numbers, and characters.

Program output:



```
Please enter start value: 35
Please enter stop value: 44
  35 = #
  36 = $
  37 = %
  38 = &
  39 = '
  40 = (
  41 = )
  42 = *
  43 = +
  44 = ,
Press any key to continue_
```

Program (`counter1.c`) *Extra Credit*

Write a program that simulates a car's trip meter.

Notice (below) that `the_ones` is initialized to 4, and `the_tens` to 9. This means that your program will reach 099 after only two passes through the loop.

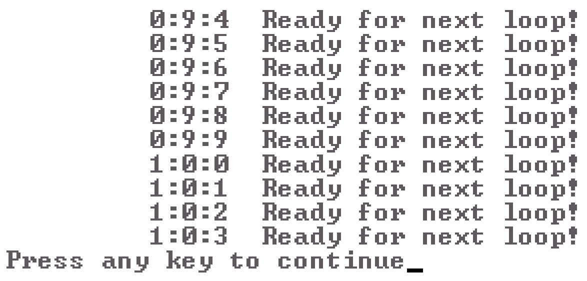
The maximum possible value for each variable is 9. When `the_ones` reaches 9, the next loop must add one to the value of `the_tens`, and set `the_ones` back to zero.

Here is how the first part of the program should be:

```
int the_ones = 4,
    the_tens = 9,
    the_hundreds = 0,
    loop_counter;

char dummy;   // for pausing the loop

  for (loop_counter = 1; loop_counter <= 10; loop_counter++)
  {
    printf("\t %d:%d:%d \t",the_hundreds,the_tens,the_ones);

    printf("Ready for next loop! ");   // pause loop
    scanf("%c",&dummy);
```

The rest of the program is the tricky part, and that's up to you.

Program output:



```
                 0:9:4  Ready for next loop!
                 0:9:5  Ready for next loop!
                 0:9:6  Ready for next loop!
                 0:9:7  Ready for next loop!
                 0:9:8  Ready for next loop!
                 0:9:9  Ready for next loop!
                 1:0:0  Ready for next loop!
                 1:0:1  Ready for next loop!
                 1:0:2  Ready for next loop!
                 1:0:3  Ready for next loop!
Press any key to continue_
```

Extra Work (Not Graded)

These programs will ***not*** be graded, but they are worth doing if you have the time.

Program (`lowest2.c`)

Write a program, similar to `highest2.c`, but it asks for 7 night time low temperatures.

The program should print out the weekly low temperature. Use a `for()` loop in this program.

Program (`even.c`)

Write a program with a `while()` loop, that asks the user for a number, and tells the user if the number is odd, or even (positive and negative should work).

The program should exit the loop when the user enters the number 9999 only.

Program (`interest1.c`)

Write a program that will calculate the compounded interest on a sum of money deposited at a fixed rate of interest over a number of years.

The program should ask for: the sum being deposited, the term of the deposit, and the rate of interest to be applied annually.

The program should calculate, and list, the value of the deposited sum at the end of each year.

Program (`ascii4.c`)

Write a program that will produce the ASCII table below:

```
 32 =         33 = !        34 = "        35 = #        36 = $
 37 = %       38 = &        39 = '        40 = (        41 = )
 42 = *       43 = +        44 = ,        45 = -        46 = .
 47 = /       48 = 0        49 = 1        50 = 2        51 = 3
 52 = 4       53 = 5        54 = 6        55 = 7        56 = 8
 57 = 9       58 = :        59 = ;        60 = <        61 = =
 62 = >       63 = ?        64 = @        65 = A        66 = B
 67 = C       68 = D        69 = E        70 = F        71 = G
 72 = H       73 = I        74 = J        75 = K        76 = L
 77 = M       78 = N        79 = O        80 = P        81 = Q
 82 = R       83 = S        84 = T        85 = U        86 = V
 87 = W       88 = X        89 = Y        90 = Z        91 = [
 92 = \       93 = ]        94 = ^        95 = _        96 = `
 97 = a       98 = b        99 = c       100 = d       101 = e
102 = f      103 = g       104 = h       105 = i       106 = j
107 = k      108 = l       109 = m       110 = n       111 = o
112 = p      113 = q       114 = r       115 = s       116 = t
117 = u      118 = v       119 = w       120 = x       121 = y
```

```
122 = z       123 = {       124 = |       125 = }       126 = ~
```