1. （將字元填塞到整數中）左移運算子可以用來將 4 個字元值填塞到一個 `unsigned` 整數裡。請撰寫一個程式由鍵盤輸入 4 個字元，然後將他們傳給函式 `packCharacters`。要將四個字元填塞到一個 unsigned 整數，首先把第一個字元設定給這個 `unsigned` 整數變數，然後把這個變數左移 8 個位元，再以位元 inclusive OR 運算子將第二個字元與這個變數結合起來。對第三和第四個字元重複進行這個步驟。你的程式應印出這 2 個字元在填塞之前和之後的位元表示格式，以驗證操作的正確性。

例如: 輸入

A: 65->0100 0001
a: 97->0110 0001
D: 68->0100 0100
i: 105->0110 1001
塞進 D 與 i 之前: 0000 0000 0000 0000 0100 0001 0110 0001

塞進 D 與 i 之後: 0100 0001 0110 0001 0100 0100 0110 1001

2. （反轉整數的位元順序）撰寫一個程式反轉一個 `unsigned` 整數值的位元順序。此程式由使用者輸入一個 `unsigned` 整數，然後呼叫函式 `reverseBits` 以相反的順序印出位元。請印出反轉前後的位元值，以驗證操作的正確性。

數字: 0000 0000 0000 0010 0100 0100 1111 1111
反轉: 1111 1111 0010 0100 0000 0000 0000 0000

1. ***(Packing Characters into an Integer)*** The left-shift operator can be used to pack four character values into a four-byte unsigned int variable. Write a program that inputs four characters from the keyboard and passes them to function packCharacters. To pack four characters into an unsigned int variable, assign the first character to the unsigned int variable, shift the unsigned int variable left by 8 bit positions and combine the unsigned variable with the second character using the bitwise inclusive OR operator. Repeat this process for the third and fourth characters. The program should output the characters in their bit format before and after they're packed into the unsigned int to prove that the characters are in fact packed correctly in the unsigned int variable.

   Example: input
   A: 65->0100 0001
   a: 97->0110 0001
   D: 68->0100 0100
   i: 105->0110 1001
   Before packet D and i: 0000 0000 0000 0000 0100 0001 0110 0001
   After packet D and i: 0100 0001 0110 0001 0100 0100 0110 1001

2. ***(Reversing the Order of an Integer's Bits)*** Write a program that reverses the order of the bits in an unsigned int value. The program should input the value from the user and call function reverseBits to print the bits in reverse order. Print the value in bits both before and after the bits are reversed to confirm that the bits are reversed properly.
   Interger: 0000 0000 0000 0010 0100 0100 1111 1111
   Reversed result: 1111 1111 0010 0100 0000 0000 0000 0000