ALAB 03/17

1. **(On Workstation) Please write a function int count(char a[], char b[]) which counts and returns the occurrences in the first argument, a, of the characters in the second argument, b. You just can use C-String functions to complete this problem. In other words, you can't use char array to implement it.**

   Ex: count("This is a abracadabra", "baxi") would return 10.

2. **(On Workstation) Please write a subroutine void replaceWord(char *arti, char *wor, char *rep), which can replace the word "wor" of article "arti" by another word "rep". Please use following code to verify your function.**

```
int main()
{
   char arti[MAX_N] = "Get free image-photo hosting, easy photo sharing, and
   photo editing. Upload picturesphoto and photovideos, create with the online
   photo editor, or browse a topphotodown gallery or photo.";
   char word[6] = "photo";
   char repl1[4] = "PEN";
   char repl2[8] = "Digital";

   printf("Original:\n%s\n\n", arti);
   replaceWord(arti, word, repl1);
   printf("Replaced:\n%s\n\n\n", arti);

   printf("Original:\n%s\n\n", arti);
   replaceWord(arti, repl1, repl2);
   printf("Replaced:\n%s\n\n\n", arti);

   return 0;
}
```

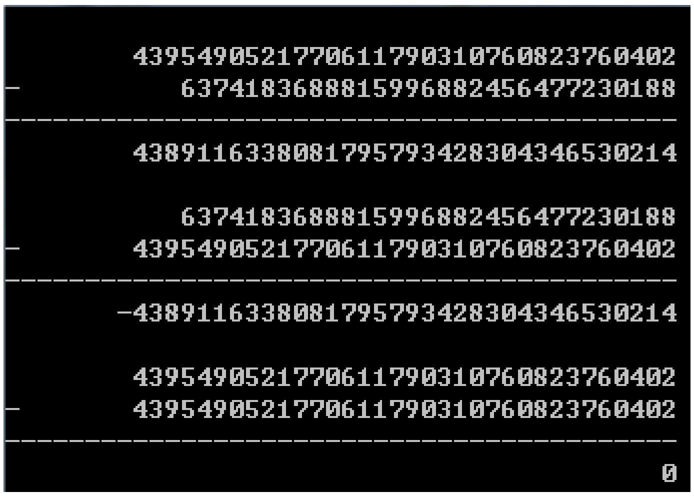
3. Subtracting Large Integers:

In C++, the largest int value is 2147483647. So an integer larger than this number cannot be stored and processed as an integer. Similarly, if the sum or product of two positive integers is greater than 2147483647, the result will be incorrect. One way to store and manipulate large integers is to

store each individual digit of the number in an array.
Write a subroutine char *sub(char *num1, char *num2) that reads two positive integers of at most 40 digits and return the difference of the numbers. Please use following code to verify your function.

```
void display(char *s1, char *s2, char *ans)
{
   printf("\n%42s\n-%41s\n",s1,s2);
   printf("------------------------------------------");
   printf("%42s",ans);
}

int main()
{
      char num1[MAX_N] = "4395490521770611790310760823760402";
      char num2[MAX_N] = "6374183688815996882456477230188";
      char *ans;

      ans = sub(num1, num2);
      display(num1, num2, ans);
      delete [] ans;

       ans = sub(num2, num1);
      display(num2, num1, ans);
      delete [] ans;

      ans = sub(num1, num1);
      display(num1, num1, ans);
      delete [] ans;

      getch();
      return 0;
}
```

Result:



```
      4395490521770611790310760823760402
-        6374183688815996882456477230188
----------------------------------------
      4389116338081795793428304346530214

         6374183688815996882456477230188
-     4395490521770611790310760823760402
----------------------------------------
     -4389116338081795793428304346530214

      4395490521770611790310760823760402
-     4395490521770611790310760823760402
----------------------------------------
                                       0
```

4. **Please design a function to check following case whether the string parameter is a palindrome or not?**

   Chinese and English mixed sentence with word as one unit and a space between Chinese word and English word.

   Ex: 林志林 love 我，我 love 林志林 -> Yes
   林志林 love 我，我 LovE 林志林 -> Yes
   林志林 love 我，我 LovE 林志穎 -> No
   林志林 love 我，我 LovE 林志林 -> Yes
   上海自來水來自海上 -> Yes
   Able saw I ere I saw able -> Yes
   ABCDE -> Yes
   我是誰 -> No
   Dog 是不是 dog -> Yes
   Dog 是不是 GOD -> No

You can refer to following code to get the bytes of UTF character occupies.

```
int UTF_8_Bytes(char headChar) {
        char mask = 0x80;
        int cnt = 1;
        if((headChar & mask) == 0) return 1;

        mask >>= 1;
        cnt++;
        while(((mask | headChar) & (mask >> 1)) != mask ) {
                mask >>= 1;
                cnt++;

                if(mask == (char)0xFE) return 1; //Error
        }
        return cnt;
}
```

5. Write a function void deleteChar(char a[], char c) which deletes any occurrence in the first argument, a, of the single character which is the second argument, c.

   Ex: a=" Thais ais an abracadabra."; c='a'; => This is n brcdbr.