MINOR ASSIGNMENT-003
Pointers & Strings in C
UNIX Systems Programming (CSE 3041)

Assignment Objectives:

To understand how a string constant is stored in an array of characters. To learn about the placeholder %s and how it is used in **`printf`** and **`scanf`** operations. To learn operations performed on strings and also operations that can be performed on individual characters. Furthermore, to understand how recursion is used as a problem solving tool.

Instruction to Students (If any):

This assignment is designed to give practice with **strings, strings processing, and array of pointers in C**. Students are required to create their own programs to solve each and every question/problem as per the specification of the given question/problem to meet the basic requirement of Unix systems programming. **Students are required to write the output/ paste the output screen shots onto their laboratory record after each question.**

Programming/ Output Based Questions:

1. We know a string in C is implemented as an array. So, declare and initialize the string **``It is very interesting''** and display the string.

2. Declare and initialize the string using array reference and pointer.

0	1	2	3	4	5	6	7	8	9	10	11
I	T	E	R		S	O	A	\0	?	?	?

3. Write a program to read a string from the keyboard and print each character with their address on the screen.

4. Create a program to display the address and value of each element of the given integer array **a**. Also perform a close observation on the format of the address and the change of address from index **0** to the last index of the array.

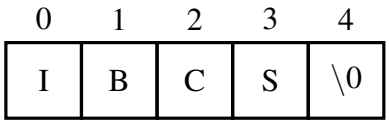
0	1	2	3	4	5	6	7	8	9
0	10	20	30	40	50	60	70	80	90
&a[0]	**&a**[1]	**&a**[2]	**&a**[3]	**&a**[4]	**&a**[5]	**&a**[6]	**&a**[7]	**&a**[8]	**&a**[9]

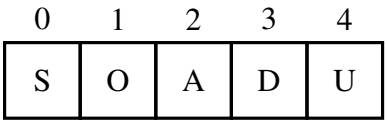
5. Declare the two arrays to hold the values as shown in the given rectangular boxes. Write the equivalent C statement to print their values and addresses through pointer.

10	13	20	33	44

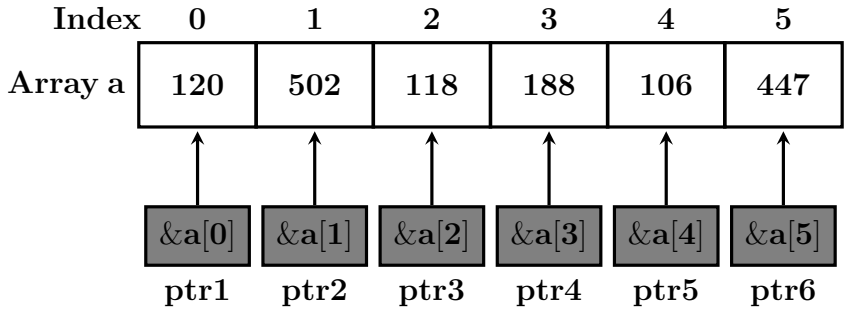
10.2	13.3	20.0	33.3	45.3	89.9

6. Declare and initialize the two arrays to hold the values as shown in the given rectangular boxes. Write the equivalent C statement to print their values and addresses using pointer.

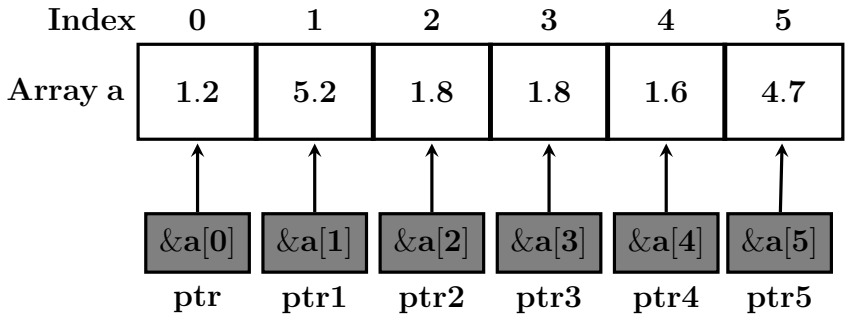




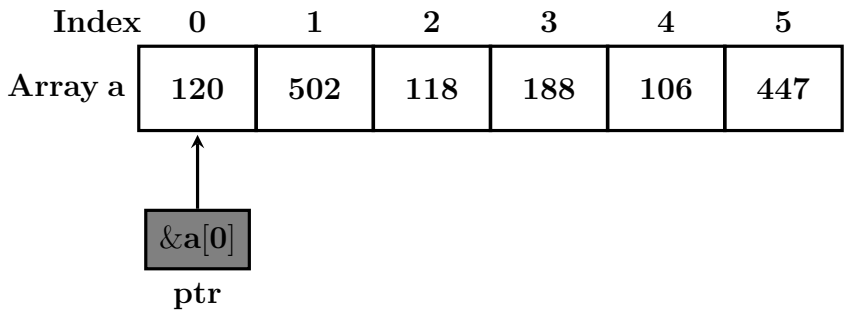
7. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using pointer.



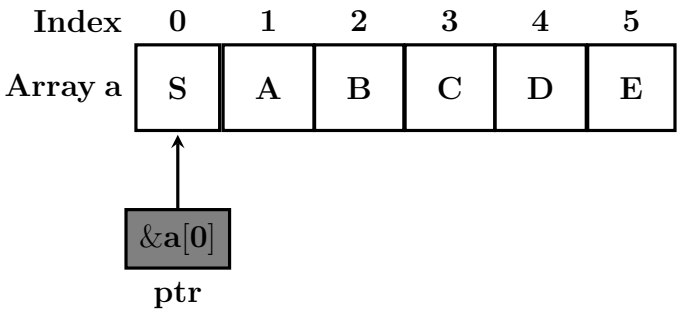
8. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using pointer.



9. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using pointer.



10. Write the C statement to declare and initialize the pointer variable for the given structure and display the array content using pointer.



11. For the given declarations **`int a[10]; int *pa;`** and assignment **`pa=a;`**, select the legal /illegal statements from the followings

```
(a) pa=a;      (f) is pa[i] identical to *(pa+i)?
(b) pa=&a[0];  (g) is &a[i] identical to (a+i)?
(c) pa++;      (h) is a[i] identical to  *(a+i)?
(d) a=pa       (i) is pa[i] identical to  a[i]?
(e) a++        (j) is void f(char str[]){...}identical to void f(char *str){...}?
(k) If a is an array, is f(&a[2]) identical to f(a+2);
```

12. Let **`p`** be a pointer to an integer array and **`n`** is a scalar value. State the significance of the statement **`p+n`**.

13. If **`arname`** is an array, the function call **`f(&arname[2]);`** passes part of an array to the function by passing a pointer to the begining of the sub-array. Write an equivalent statement for the call **`f(&arname[2]);`**.

14. Write an equivalent statement for the function's formal parameter **`a`**, whose header/definition is given as **`f(int a[], int n, float y)`** {`.....`}.

15. Find the output of the following code segment for the function call **`bc=bytescount(``COVID-19 Still Active'')`**

```
int bytescount(char *s){
   char *p=s;
   while(*p!='\0'){
       p++;
   }
   return p-s;
}
```

16. Find the output of the following code segment for the function call **`cc=countchar(``Encourged to Vaccinate'')`**

```
int countchar(char *s){
   int n;
   for(n=0; *s!='\0'; s++){
       n++;
   }
   return n;
}
```

17. Identify the type of variable **amsg** and **pmsg** from the following declaration and initialization statements.

```
char pmsg[]="I am in 5th Sem CSE";
char *amsg="I am in 5th Sem CSE";
```

18. Find the output of the code fragment

```
char pmsg[60];
int nc;
nc=charcopy(pmsg,"I am in 5th Sem CSE");
printf("%d...%s\n",nc,pmsg);
```

The function definition/header is given as

```
int  charcopy(char *s, char *t)
{
   int i=0;
   while((s[i]=t[i])!='\0')
         i++;
   s[i]='\0';
   return(i);
}
```

19. The function header is given as;

```
int  charcopy(char *s, char *t){
  int i=0;
  while((*s=*t)!='\0'){
        s++;
        t++;
        i++;
  }
  *s='\0';
  return(i);
}
```

Compute the output of the following code segment

```
char pmsg[60];
int nc;
nc=charcopy(pmsg,"Studied in CSE
   ");
printf("%d...%s\n",nc,pmsg);
```

20. The function header is given as;

```
int  charcopy(char *s, char *t){
   int i=0;
   while((*s++=*t++)!='\0'){
         i++;
   }
   *s='\0';
   return(i);
}
```

Compute the output of the following code segment

```
char pmsg[60];
int nc;
nc=charcopy(pmsg,"ITER CSE ");
printf("%d...%s\n",nc,pmsg);
```

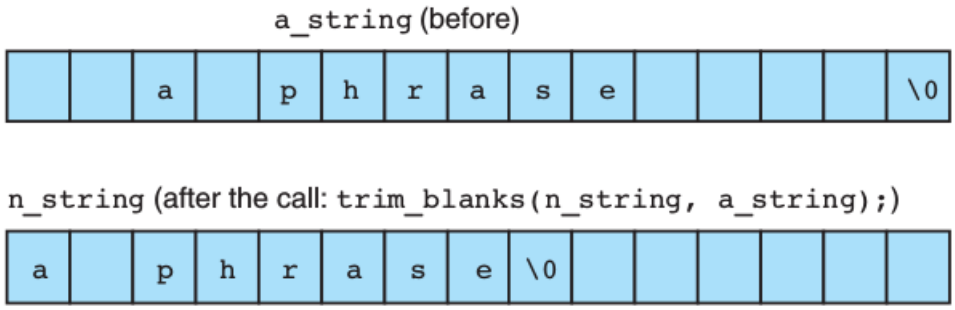
21. Write a pointer version of string concatenation program using the user-defined function, **`stringconcate(s,t);`**, copies the string **`t`** to the end **`s`**.

22. Write your own versions of the library functions **`strncpy`**, **`strncat`**, and **`strncmp`** which operate on at most the first **`n`** characters of their argument strings. For example **`strncpy(s,t,n)`** copies at most **`n`** characters of **`t`** to **`s`**.

23. Write a program to take a product code from Millies Mail-Order Catalog (MMOC) and separate it into its component parts. An MMOC product code begins with one or more letters identifying the warehouse where the product is stored. Next come the one or more digits that are the product ID. The final field of the string starts with a capital letter and represents qualifiers such as size, color, and so on. For example, ATL1203S14 stands for product 1203, size 14, in the Atlanta warehouse. Write a program that takes a code, finds the position of the first digit and of the first letter after the digits, and uses **`strcpy`** and **`strncpy`** to display a report such as the following:

```
Warehouse: ATL
Product: 1203
Qualifiers: S14
```

24. Complete function **`trim_blanks(...)`** whose purpose is to take a single string input parameter (**`to_trim`**) and return a copy of the string with leading and trailing blanks removed. Use **`strncpy`** in **`trim_blanks`**.




```
char * trim_blanks(char *trimmed,                    /* output */
                   const char *to_trim)  /* input */
{
     /* Find subscript of first nonblank in to_trim */
     /* Find subscript of last nonblank in to_trim */
    /* Use strncpy to store trimmed string in trimmed */
}
```

25. Draw an array to show the output of the function, **`strcat(s1, s2)`**, used in the following code snippet and also write the output.

```
#define STRSIZ 20
char s1[STRSIZ]="Jupiter ", s2[STRSIZ]="Symphony";
printf("%d %d\n", strlen(s1), strlen(strcat(s1, s2)));
printf("%s\n", s1);
```

26. Given the string **pres** (value is **"Adams, John Quincy"**) and the 40-character temporary variables **tmp1** and **tmp2**, what string is displayed by the following code fragment?

```
strncpy(tmp1, &pres[7], 4);
tmp1[4] = '\0';
strcat(tmp1, " ");
strncpy(tmp2, pres, 5);
tmp2[5] = '\0';
printf("%s\n", strcat(tmp1, tmp2));
```

27. Write a program to check a string is palindrome or not. For example, **madam** is a palindrome, **computer** is not a palindrome.

28. Write a program in C to input a string using **getchar()** function only (Do not use **scanf()** or **gets()** function) and then count the total number of alphabets, number of alphabets in uppercase, number of alphabets in lowercase, number of digits, number of punctuation symbols, and number of spaces using character library functions.

Sample Run:

```
Input a string: I'm 2 bz 4 now.
Total number of alphabets: 7
Number of uppercase alphabets: 1
Number of lowercase alphabets: 6
Number of digits: 2
Number of punctuation mark: 2
Number of spaces: 4
```

29. Write a program in C to read N strings from user and then sort them using bubble sort.

Sample Run:

```
Input number of strings :3
Input 3 strings:
hello
world
fun
The sorted strings are:
fun
hello
world
```

30. Write a function **bracket_by_len** that takes a word as an input argument and returns the word bracketed to indicate its length. Words less than five characters long are bracketed with $<<\quad>>$, words five to ten letters long are bracketed with $(*\quad*)$, and words over ten characters long are bracketed with $/+\quad+/$. Your function should require the calling function to provide as the first argument, space for the result, and as the third argument, the amount of space available. Consider the expected results of these calls to the function.

Sample Run:

```
CASE-1
       Call: bracket_by_len(tmp, "insufficiently", 20);
       output:          "/+insufficiently+/"
CASE-2
      Call: bracket_by_len(tmp, "the", 20)
      output:           "<<the>>"
```

31. What is the value of **t1** after execution of these statements if the value of **t2** is ``**Merry Christmas**''?

```
strncpy(t1, &t2[3], 5);
t1[5] = '\0';
```

32. What does this program fragment display?

```
char x[80] = "gorilla";
char y[80] = "giraffe";
strcpy(x, y);
printf("%s %s\n", x, y);
```

33. What does this program fragment display?

```
char x[80] = "gorilla";
char y[80] = "giraffe";
strcat(x, y);
printf("%s %s\n", x, y);
```