Tutorial 4 – Character Strings

1. What does the following program print?

```
#include  <stdio.h>
#include  <string.h>
#define  M1  "How are ya, sweetie?"
char M2[40] = "Beat the clock.";
char *M3 = "chat";

int main()
{
    char words[80],*p;
    printf(M1);
    puts(M1);
    puts(M2);
    puts(M2+1);
    fgets(words, 80, stdin);    /* user inputs :  win a toy. */
    if (p=strchr(words,'\n')) *p = '\0';
    puts(words);
    scanf("%s", words+6);    /* user inputs :  snoopy. */
    puts(words);
    words[3] = '\0';
    puts(words);
    while (*M3)  puts(M3++);
    puts(--M3);
    puts(--M3);
    M3 = M1;
    puts(M3);
    return 0;
}
```

2. The following unknown function receives a string argument and a character argument, modifies the string argument and returns an integer value. Describe the purpose of the function. Give an example to support your answer.

```
int unknown(char str[ ], char c)
{
    int x, y=0, z=0;
    for (x=0; str[x] != '\0'; x++)
        if (str[x] != c)
            str[y++] = str[x];
        else
            z++;
    str[y] = '\0';
    return z;
}
```

char c is o, char str[] is potato

y=ptat

z=2

z counts the number of times char c appears in str[]
y is the string without all instances of char c .

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
```

3. Write the function strncpy() that copies not more than n characters (characters that follow a null character are not copied) from the array pointed to by s2 to the array pointed to by s1. If the array pointed to by s2 is a string shorter than n characters, null characters are appended to the copy in the array pointed to by s1, until n characters in all have been written. The strncpy returns the value of s1. The function prototype is:

   ```
   char *strncpy(char * s1, char * s2, int n);
   ```

   Write a C program to test the function.

   Some sample input and output sessions are given below:

   (1) Test Case 1
   Enter the string:
   I am a boy.
   Enter the number of characters:
   7
   stringncpy(): I am a

   (2) Test Case 2
   Enter the string:
   I am a boy.
   Enter the number of characters:
   21
   stringncpy(): I am a boy.

```
int main(){

   int number;
   char end[80],*p;
   char source[80], *t;

   printf("Enter a string: \n");
   fgets(source,80,stdin);
   if(t=strchr(source,'\n')) *t ='\0';
   printf("%s",source);

   int n,i;
   printf("Enter the number of
characters: \n");
   scanf("\n");
   scanf("%d", &n);
   source[n] = '\0';

   printf("stringncpy():\n");
   printf("%s",source);
}
```

4. Write a C function that compares the string pointed to by *s1* to the string pointed to by *s2*. If the string pointed to by *s1* is greater than, equal to, or less than the string pointed to by *s2*, then it returns 1, 0 or –1 respectively. Write the code for the function without using the standard C string library function strcmp(). The function prototype is given as follows:

   ```
   int stringcmp(char *s1, char *s2);
   ```

   Write a C program to test the function.

   Some sample input and output sessions are given below:

   (1) Test Case 1:
   Enter a source string:
   abc
   Enter a target string:
   abc
   stringcmp(): equal

   (2) Test Case 2:
   Enter a source string:
   abcdefg
   Enter a target string:
   abcde123

letter + digit

stringcmp(): greater than

(3) Test Case 3:
Enter a source string:
abc123
Enter a target string:
abcdef
stringcmp(): less than

(4) Test Case 4:
Enter a source string:
abcdef
Enter a target string:
abcdefg
stringcmp(): less than

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>

int stringcmp(char*s1,char*s2);

int main(){

   int i;
   char target[80],*p;
   char source[80], *t;
   int result = 0;

   printf("Enter a source string: \n");
   fgets(source,80,stdin);
   if(t=strchr(source,'\n')) *t ='\0';

   printf("Enter a target string: \n");
   fgets(target,80,stdin);
   if(p=strchr(target,'\n')) *p='\0';

   result = stringcmp(source,target);
   printf("stringcmp(): ");
   if (result == 0)
   {
      printf("equal");
   }
   else if (result == 1)
   {
      printf("greater than");
   }
   else
   {
      printf("less than");
   }

}

int stringcmp(char *s1, char *s2)
{
   int compare = 0;

   while((*s1 != '\0' && *s2 != '\0')&& *s1 == *s2)        //while portion
gets to the first different character
   {
      s1++;
      s2++;
   }
   //compare = (*s1 == *s2) ? 0 : (*s1 >*s2) ? 1: -1;

   if (*s1 == *s2)
   {
      compare = 0;
   }
   else if (*s1 >*s2)
   {
      compare = 1;
   }
   else
   {
      compare = -1;
   }

   printf("%d",compare);
   return compare;

}
```