Week 5 Lab: Character Strings – Suggested Solutions

Q1 (sweepSpace)

```c
#include <stdio.h>
#include <string.h>
char *sweepSpace1(char *str);
char *sweepSpace2(char *str);
int main()
{
   char str[80], *p;

   printf("Enter the string: \n");
   fgets(str, 80, stdin);
   if (p=strchr(str,'\n')) *p = '\0';
   printf("sweepSpace1(): %s\n", sweepSpace1(str));
   printf("sweepSpace2(): %s\n", sweepSpace2(str));
   return 0;
}
char *sweepSpace1(char *str)
{
   int i, j, len;

   i=0; len=0;
   while (str[i]!='\0'){
      len++;
      i++;
   }
   j = 0;
   for ( i=0; i < len; i++)
   {
      if (str[i] != ' ')
      {
         str[j] = str[i];
         j++;

      }
   }
   str[j] = '\0';
   return str;
}
char *sweepSpace2(char *str)
{
   int i, j, len;

   i=0; len=0;
   while (*(str+i)!='\0'){
      len++;
      i++;
   }
   j = 0;
   for ( i=0; i < len; i++)
   {
      if (*(str+i) != ' ')
      {
         *(str+j) = *(str+i);
         j++;
      }
   }
   *(str+j) = '\0';
   return str;
```

```
}
```

Q2 (findTarget)

```
#include <stdio.h>
#include <string.h>
#define SIZE 10
#define INIT_VALUE 999
void printNames(char nameptr[][80], int size);
void readNames(char nameptr[][80], int *size);
int findTarget(char *target, char nameptr[][80], int size);
int main()
{
   char nameptr[SIZE][80], t[40], *p;
   int size, result = INIT_VALUE;
   int choice;

   printf("Select one of the following options: \n");
   printf("1: readNames()\n");
   printf("2: findTarget()\n");
   printf("3: printNames()\n");
   printf("4: exit()\n");
   do {
      printf("Enter your choice: \n");
      scanf("%d", &choice);
      switch (choice) {
         case 1:
            readNames(nameptr, &size);
            break;
         case 2:
            printf("Enter target name: \n");
            scanf("\n");
            fgets(t, 80, stdin);
            if (p=strchr(t,'\n')) *p = '\0';
            result = findTarget(t, nameptr, size);
            printf("findTarget(): %d\n",  result);
            break;
         case 3:
            printNames(nameptr, size);
            break;
      }
   } while (choice < 4);
   return 0;
}
void printNames(char nameptr[][80], int size)
{
   int i;

   for (i=0; i<size; i++)
      printf("%s ", nameptr[i]);
   printf("\n");
}
void readNames(char nameptr[][80], int *size)
{
   int i;

   printf("Enter size: \n");
   scanf("%d", size);
   printf("Enter %d names: \n", *size);
   for (i=0; i < *size; i++)
      scanf("%s", nameptr[i]);
```

```
}
int findTarget(char *target, char nameptr[][80], int size)
{
   int i;
   for (i=0; i<size; i++) {
      if (strcmp(nameptr[i], target) == 0)
         return i;
   }
   return -1;
}
```

Q3 (palindrome)

```
#include <stdio.h>
#include <string.h>
#define INIT_VALUE -1000
int palindrome(char *str);
int main()
{
   char str[80], *p;
   int result = INIT_VALUE;

   printf("Enter a string: \n");
   fgets(str, 80, stdin);
   if (p=strchr(str,'\n')) *p = '\0';
   result = palindrome(str);
   if (result == 1)
      printf("palindrome(): A palindrome\n");
   else if (result == 0)
      printf("palindrome(): Not a palindrome\n");
   else
      printf("An error\n");
   return 0;
}
int palindrome(char *str)
{
   int len, i;
   char *p1, *p2;

   i=0; len=0;
   while (*(str+i)!='\0') {
      i++;
      len++;
   }
   p1=str;
   p2=str+len-1;
   while (p1<p2){
      if (*p1 != *p2)
         break;
      else {
         p1++;
         p2--;
      }
   }
   if (p1<p2)
      return 0;
   else
      return 1;
}
```