

```

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

char TempStr[1000];
typedef struct strholder
{
    struct strholder *Next;
    char *Value;
}
HOLDER;

void main()
{
    char **MyString;
    int Count,i;
    HOLDER *TempHold,*Head,*TempHold2;

    // Initialization
    Count = 0;
    Head = NULL;
    printf("Enter a string: ");
    scanf("%s",TempStr);
    // Read a list of strings
    while (!feof(stdin))
    {
        // Allocate the chaining structure
        TempHold = (HOLDER *)malloc(sizeof(HOLDER));
        if (TempHold == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        // Allocate space for the string & store it
        TempHold->Value = (char *)malloc(strlen(TempStr)+1);
        if (TempHold->Value == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        strcpy(TempHold->Value,TempStr);
        // chain the string
        TempHold->Next = Head;
        Head = TempHold;
        // count the string
        Count++;
        // Read the next
        printf("Enter a string: ");
        scanf("%s",TempStr);
    }
    // Allocate the array
    MyString = (char **)calloc(Count,sizeof(char *));
    if (MyString == NULL)
    {
        fprintf(stderr,"Memory Allocation Failed\n");
        exit(100);
    }
    // Copy Strings to array
    for (TempHold = Head,i=Count-1 ; TempHold ; TempHold = TempHold->Next,i--)
    {
        MyString[i] = TempHold->Value;
    }
    // Process the array
    for (i=0 ; i<Count ; i++)
    {
        printf("%d You Entered %s\n",i,MyString[i]);
    }
    // free the list and the strings
    for (TempHold = Head ; TempHold ; TempHold = TempHold->Next)
    {
        TempHold2 = TempHold->Next;
        free(TempHold->Value);
        free(TempHold);
    }
}

```

```
}  
// free the array  
free(MyString);  
}
```

```

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

char TempStr[1000];
typedef struct strholder
{
    struct strholder *Next;
    char *Value;
}
HOLDER;

void main()
{
    char **MyString;
    int Count,i;
    HOLDER *TempHold,*Head,*Tail,*TempHold2;

    // Initialization
    Count = 0;
    Head = NULL;
    Tail = NULL;
    printf("Enter a string: ");
    scanf("%s",TempStr);
    // Read a list of strings
    while (!feof(stdin))
    {
        // Allocate the chaining structure
        TempHold = (HOLDER *)malloc(sizeof(HOLDER));
        if (TempHold == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        // Allocate space for the string & store it
        TempHold->Value = (char *)malloc(strlen(TempStr)+1);
        if (TempHold->Value == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        strcpy(TempHold->Value,TempStr);
        // chain the string
        if (Head == NULL)
        {
            Head = TempHold;
        }
        else
        {
            Tail->Next = TempHold;
        }
        TempHold->Next = NULL;
        Tail = TempHold;
        // count the string
        Count++;
        // Read the next
        printf("Enter a string: ");
        scanf("%s",TempStr);
    }
    // Allocate the array
    MyString = (char **)calloc(Count,sizeof(char *));
    if (MyString == NULL)
    {
        fprintf(stderr,"Memory Allocation Failed\n");
        exit(100);
    }
    // Copy Strings to array
    for (TempHold = Head,i=0 ; TempHold ; TempHold = TempHold->Next,i++)
    {
        MyString[i] = TempHold->Value;
    }
    // Process the array
    for (i=0 ; i<Count ; i++)

```

```
{
    printf("%d You Entered %s\n",i,MyString[i]);
}
// free the list and the strings
for (TempHold = Head ; TempHold ; TempHold = TempHold2)
{
    TempHold2 = TempHold->Next;
    free(TempHold->Value);
    free(TempHold);
}
// free the array
free(MyString);
}
```

```

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

char TempStr[1000];
typedef struct strholder
{
    struct strholder *Next;
    char *Value;
}
HOLDER;

void main()
{
    int Count;
    HOLDER *TempHold,*Head,*Tail,*TempHold2,*Prev;

    // Initialization
    Count = 0;
    Head = NULL;
    Tail = NULL;
    printf("Enter a string: ");
    scanf("%s",TempStr);
    // Read a list of strings
    while (!feof(stdin))
    {
        // Allocate the chaining structure
        TempHold = (HOLDER *)malloc(sizeof(HOLDER));
        if (TempHold == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        // Allocate space for the string & store it
        TempHold->Value = (char *)malloc(strlen(TempStr)+1);
        if (TempHold->Value == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        strcpy(TempHold->Value,TempStr);
        // chain the string
        if (Head == NULL)
        {
            Head = TempHold;
        }
        else
        {
            Tail->Next = TempHold;
        }
        TempHold->Next = NULL;
        Tail = TempHold;
        // count the string
        Count++;
        // Read the next
        printf("Enter a string: ");
        scanf("%s",TempStr);
    }
    // Delete the word "red" (first occurrence only)
    // first, find the word
    for (TempHold = Head,Prev = NULL;
        TempHold && strcmp("red",TempHold->Value) != 0;
        Prev=TempHold,TempHold=TempHold->Next);
    // check to see if we found it
    if (TempHold != NULL)
    {
        // remove the item from the list
        if (Prev == NULL)
        {
            Head = TempHold->Next;
        }
        else
        {

```

```

    Prev->Next = TempHold->Next;
}
// adjust the tail pointer, in case we want to add more
if (TempHold->Next == NULL)
{
    Tail = Prev;
    // have we deleted the only item in the list?
    if (Tail == NULL)
    {
        Head = NULL;
    }
}
// adjust the count
Count--;
}
// Process the list
for (TempHold = Head ; TempHold ; TempHold = TempHold->Next)
{
    printf("You Entered %s\n",TempHold->Value);
}
// free the list and the strings
for (TempHold = Head ; TempHold ; TempHold = TempHold2)
{
    TempHold2 = TempHold->Next;
    free(TempHold->Value);
    free(TempHold);
}
}

```

```

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

char TempStr[1000];
typedef struct strholder
{
    struct strholder *Next;
    struct strholder *Prev;
    char *Value;
}
HOLDER;

void main()
{
    int Count;
    HOLDER *TempHold,*Head,*Tail,*TempHold2;

    // Initialization
    Count = 0;
    Head = NULL;
    Tail = NULL;
    printf("Enter a string: ");
    scanf("%s",TempStr);
    // Read a list of strings
    while (!feof(stdin))
    {
        // Allocate the chaining structure
        TempHold = (HOLDER *)malloc(sizeof(HOLDER));
        if (TempHold == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        // Allocate space for the string & store it
        TempHold->Value = (char *)malloc(strlen(TempStr)+1);
        if (TempHold->Value == NULL)
        {
            fprintf(stderr,"Memory Allocation Failed\n");
            exit(100);
        }
        strcpy(TempHold->Value,TempStr);
        // chain the string
        if (Head == NULL)
        {
            Head = TempHold;
            TempHold->Prev = NULL;
        }
        else
        {
            Tail->Next = TempHold;
            TempHold->Prev = Tail;
        }
        TempHold->Next = NULL;
        Tail = TempHold;
        // count the string
        Count++;
        // Read the next
        printf("Enter a string: ");
        scanf("%s",TempStr);
    }
    // Delete the word "red" (first occurrence only)
    // first, find the word
    for (TempHold = Head;
        TempHold && strcmp("red",TempHold->Value);
        TempHold=TempHold->Next);
    // check to see if we found it
    if (TempHold != NULL)
    {
        // remove the item from the list
        if (TempHold->Prev == NULL)
        {
            Head = TempHold->Next;

```

```

    }
else
{
    TempHold->Prev->Next = TempHold->Next;
}
if (TempHold->Next == NULL)
{
    Tail = TempHold->Prev;
}
else
{
    TempHold->Next->Prev = TempHold->Prev;
}
// adjust the count
Count--;
}
// Process the list
for (TempHold = Head ; TempHold ; TempHold = TempHold->Next)
{
    printf("You Entered %s\n",TempHold->Value);
}
// free the list and the strings
for (TempHold = Head ; TempHold ; TempHold = TempHold2)
{
    TempHold2 = TempHold->Next;
    free(TempHold->Value);
    free(TempHold);
}
}
}

```