When we allocate a memory by freeing(double) then we get error as

Double free detected

**STRING**

* String is a collection of characters with **null character.**
* Character does not end up with the null character
* Null is used because the final point will not be known – it can accept n number of characters
* String cannot be given without size (col size is necessary not row size)
* When passing a function we need to pass only the name of the array not the address because the usage &\* will nullify.

String can be denoted in various ways

char Name[6] = {'H','e','l','l','o','\0'};

char Name[6] = {"Hello"};

char greet[] = {"Hello World"};

Static declaration - char str1[21];

-char Name[][20]; ----- this denotes that any number of rows can be added to this colimn where the character of the string must not exceed the size i.e 19

-char names[10][50]; ---- there are 10 names with each having maximum capacity of 50(including null at end)

// Online C compiler to run C program online

/\*

char a = 'C';

char Name[5] = {'H','e','l','l','o'};

int arr[5] = {1,2,3,4,5};

for(i=0;i<5;i++)

printf("%d",arr);

for(i=0;i<5;i++)

printf("%c",Name);

char Name[6] = {'H','e','l','l','o','\0'};

char Name[6] = {"Hello"};

char greet[] = {"Hello World"};

char str1[21];

char Name[][20];

\*/

#include <stdio.h>

int main()

{

char Name[5];

int i;

for(i=0;i<5;i++)

scanf("%c",&Name[i]);

for(i=0;i<5;i++)

printf("\n%c=%d",Name[i],Name[i]);

printf("\n\n");

return 0;

}

OUTPUT

Chinmayee

C=67

h=104

i=105

n=110

m=109

Importance of null character

When printf is used it will be not knowing where to stop, it just goes until null value is reached so the output is fetched with the illegal char also but with stack smashing detected error

#include <stdio.h>

int main()

{

char Name[5];

int i;

for(i=0;i<5;i++)

scanf("%c",&Name[i]);

Name[5-1] = '\0';

for(i=0;i<5;i++)

printf("\n%c=%d",Name[i],Name[i]);

printf("\n\n");

printf("\n%s",Name);

puts(Name);

printf("\n\n");

return 0;

}

OUTPUT

/tmp/qbObETkAXq.o

Chinmayee

C=67

h=104

i=105

n=110

**strcpy**

First argument should be an empty string

* Null character is added by the strcpy at destination
* Strncpy to copy specific number of characters and can also send the base address

**Strcat**

It adds source to the destination

* Strncat concats the specific first characters with the destination
* Equal number of length for destination and source cannot be accepted because it already needs one extra space for null character

**Strcmp**

It checks char by char and by using ASCII values

Ex : str1 – Chinnu

Str2 - Chinm

It stops at the point when it is uncomparable i.e at n

* It gives output in integer – is s1<s2 --- negative value

s1>s2 --- positive value

s1 = s2 ---- 0

As per the above example we get negative value because the ASCII value of n is less than m

**Note**: memset, memcpy --- to initialize strucures

--To set strings

Strlen – gives number of characters present in string excluding the null character

Strerror – capture string handling errors

Strstr – to find the first occurrence of substring in main string

* These points to the beginning of the located subdirectory if not found returns null

Strchr – to find the first occurrence of character and gives base address of the same

Strrchr - to find the last occurrence of character and gives base address of the same

Strtok – exams are based on this

Csv files store the records with the comma separator – we use strtok to get different piece of arrays with the help of delimiter(temporary) to fetch all the character from there

-Only the index values varies due to the delimiter without affecting the original string

#include <stdio.h>

#include <string.h>

int main()

{

char s1[20],s2[20];

int ret = 0;

scanf("%s%s",s1,s2);

printf("\ns1=%s\ts2=%s",s1,s2);

ret = strcmp(s1,s2);

printf("Ret = %d",ret);

return 0;

}

OUTPUT

Chin

Chinnu

s1=Chin s2=ChinnuRet = -110

d1 and d2 are static array so address will already be assigned by compiler so it cannot change its address

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

char s1[20],s2[20];

char d1[40],d2[40];

char \*ptr = NULL;

int ret = 0;

scanf("%s%s",s1,s2);

printf("\ns1=%s\ts2=%s",s1,s2);

//ptr = (char \*)malloc(strlen(s1)+1);

/\*

ret = strcmp(s1,s2);

\*/

ptr = strcpy(d1,s1);

printf("\nd1=%s",d1);

printf("\nptr=%s",ptr);

return 0;

}

OUTPUT

chin

Chin

s1=chin s2=Chin

d1=chin

ptr=chin

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

char s1[20],s2[20];

char d1[40],d2[40];

char \*ptr = NULL;

scanf("%s%s",s1,s2);

strcpy(d1,s1);

printf("\nd1=%s",d1);

strcat(d1,s2);

printf("\nd1=%s",d1);

printf("\n\n");

return 0;

}

OUTPUT

Chinmayee

Murthy

d1=Chinmayee

d1=ChinmayeeMurthy

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main()

{

char s1[400],s2[20];

char \*ptr = NULL;

printf("\nEntera line with delimiter \n");

scanf("%[^\n]s",s1);

ptr = strtok(s1,",");

printf("\nptr = %s",ptr);

printf("\n\n");

return 0;

}

O/P

Entera line with delimiter

Chinnu,Chinmayee

ptr = Chinnu

printf("\nEnter a line with delimiter \n");

scanf("%[^\n]s",s1);

ptr = strtok(s1,",");

printf("\nptr = %s",ptr);

ptr = strtok(NULL,",");

printf("\nptr = %s",ptr);

O/P

Enter a line with delimiter

chin,rag

ptr = chin

ptr = rag

If we use s1,”,” 2nd time we get the next string as well

printf("\nEnter a line with delimiter \n");

scanf("%[^\n]s",s1);

ptr = strtok(s1,",");

printf("\nptr = %s",ptr);

ptr = strtok(NULL,",");

printf("\nptr = %s",ptr);

ptr = strtok(NULL,",");

printf("\nptr = %s",ptr);

ptr = strtok(NULL,",");

if(ptr == NULL)

printf("\nEnd of string");

else

printf("\nptr = %s",ptr);

O/P

Enter a line with delimiter

Chin,rag,sav

ptr = Chin

ptr = rag

ptr = sav

End of string

Assignment

To reverse a given string

To reverse a given intervals of a string

// Online C compiler to run C program online

#include <stdio.h>

#include <string.h>

int main()

{

char str[100];

printf("Enter a string: ");

fgets(str, sizeof(str),stdin);

strtok(str, "\n");

reverseString(str);

printf("Reversed string: %s\n",str);

return 0;

}

void reverseString(char str[])

{

int start = 0;

int end = strlen(str) - 1;

while(start < end)

{

char temp = str[start];

str[start] = str[end];

str[end] = temp;

start++;

end--;

}

}

// Online C compiler to run C program online

#include <stdio.h>

#include <string.h>

int main()

{

char str[100];

int start,end;

printf("Enter a string: ");

fgets(str, sizeof(str),stdin);

strtok(str, "\n");

printf("Enter the index: ");

scanf("%d",&start);

reverseInterval(str,start,end);

printf("Modified string: %s\n",str);

return 0;

}

void reverseInterval(char str[],int start, int end)

{

if(start >= 0 && end <= strlen(str) && start < end)

{

while(start < end)

{

char temp = str[start];

str[start] = str[end];

str[end] = temp;

start++;

end--;

}

}

else

{

printf("\nInvalid Interval\n");

}

}

Check once again

USER DEFINED DATATYPES

Structures

Unions

Enums

struct, union, enum

struct tagName

{

members of the structure

};

struct Square

{

int len;

int breadth;

};

struct Chair

{

int leg;

char make[20];

char material[20];

char colour[20];

float price;

char DOM[20];

};

int a1,a2;

struct Square s1, s2, s3;

arrays of structures can also be made s[10]

-It can also be declared after }; but this makes them as fixed so it is better to follow the above format

To specify user defined datatype

----- typedef struct Square SQR; (Capital letters denotes that it is structures)

SQR s1, s2, s3; OR

Typedef struct Square

{

int len;

int breadth;

}SQR1; ---- it behaves like alias name

Members/Properties/Elements can be accessed by using

* SQR1 s1;

1. . 🡺 is used when it is a static variable (nameVar.memberName)
2. -> 🡺 can be used when it is a pointer variable (nameVar -> memberName)

It should be read from right to left

* SQR1 \*ptr

Ptr -> len

Void func(struct Square)

**Note**: A structure cannot have functions

* Define structure globally (in header file i.e above the main)

Structure padding --- byte get wasted when randomly declared

To avoid structure padding ----- All numerics have to be declared at the beginning and characters at the end or vise versa

Ex: int eId;

float esal;

char eName[20];

char eGender;

char eAddress[20];

SRS

WAP to which should have the following functionalities

1. Scan employee recs
2. Display
3. Search for a particularly employee based on
4. Id
5. lName
6. search for employees whose salary fall in between 6-10K and display only those records
7. update employee rec to change his/her lName