C_98 ⇒ Return a String from a function

* Function will return but we dont pass argument

* How to return "Jerry"

* Discuss possible ways to return string with different examples.

Example ① Step1

```
void main()
{ display() = str;
}
```

Char *str;

Date
Page

? display()

```
{
  return "Jenny";
}
```

Main

Step 2

```
Char * display();
void main()
{
   Char * str;
   str = display();
   printf ("string is %s", str);
}
```

Str
100

```
Char * display()
{
   return "Jenny";
}
```

Return type ←

display()

Str

J | E | N | N | Y

100 101 ----

* We are not returning the entire "Jenny"; instead we return address of str

String is character array; so in return type we cannot char instead we write pointer to first character to string (char *).

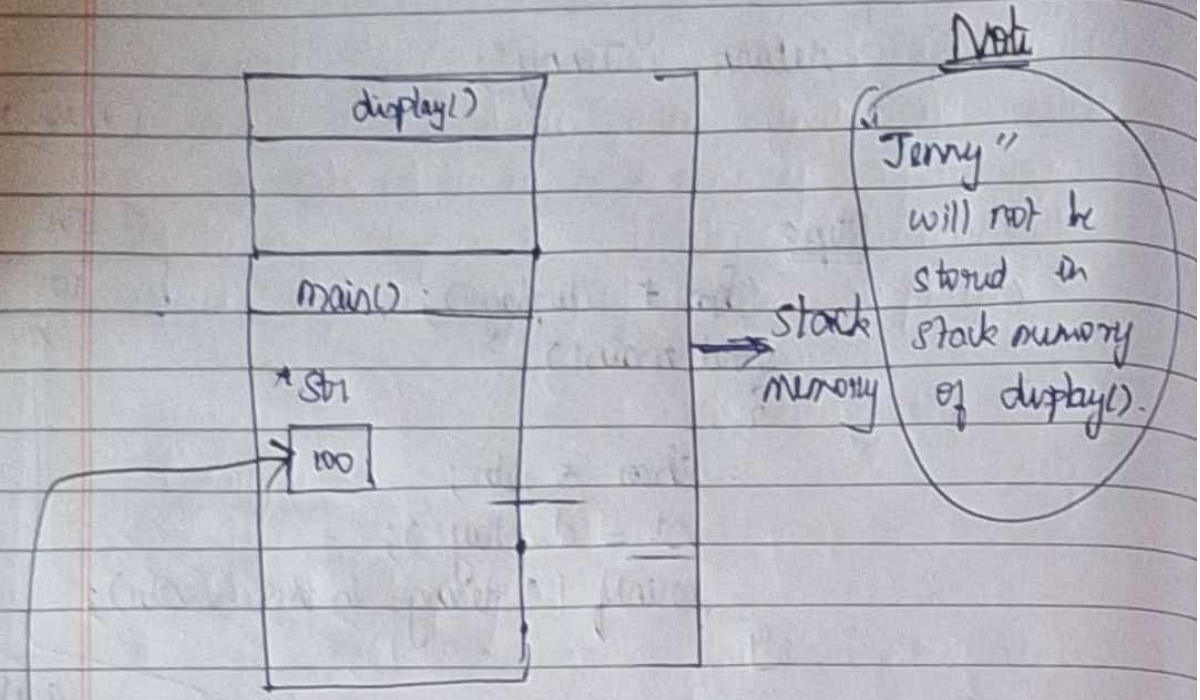
Try

Example ②

```
Char* display()
{
  Char str[] = "Jenny";
  return str;
}
```

Date
Page

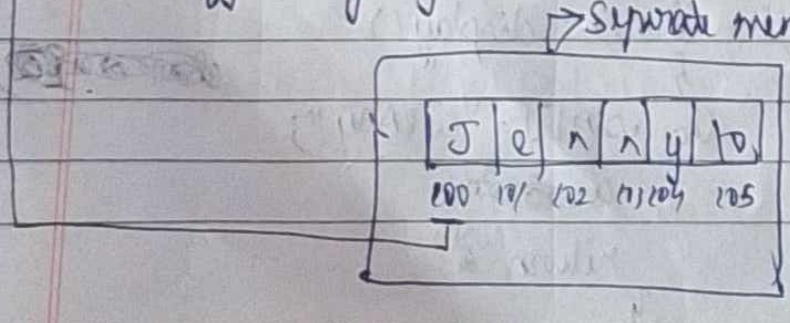
Example ① Explanation



* But if memory allocated in heap memory it wont be destroyed even after the function lifetime over.

* So we use dynamic memory allocation method; to allocate memory in heap.

* So now "jerry" will be static memory; so it will be valid through out entire program but it is only read only memory; hence we cannot modify "jerry".



Example ② Explanation

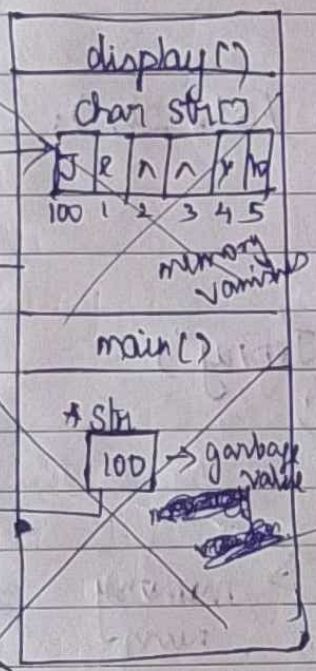
```
char * display();

void main()
{
    char *str;
    str = display();
    printf("String is :%s", str);
}

char* display()
{
    char str[] = "Jenny";
    return str;
}
```

goes to main

Cannot be accessed & shows garbage value.

display() memory vanishes

Control out from display.



Note

* Here "Jenny" will be stored in heap memory of display().

Note: (step 1)

* So; we print the string which is initialized in called function but accessed and printed in main function

(Step 2)

But this program won't work because; once the control come out from display() char str[] is vanished and cannot be accessed in main.

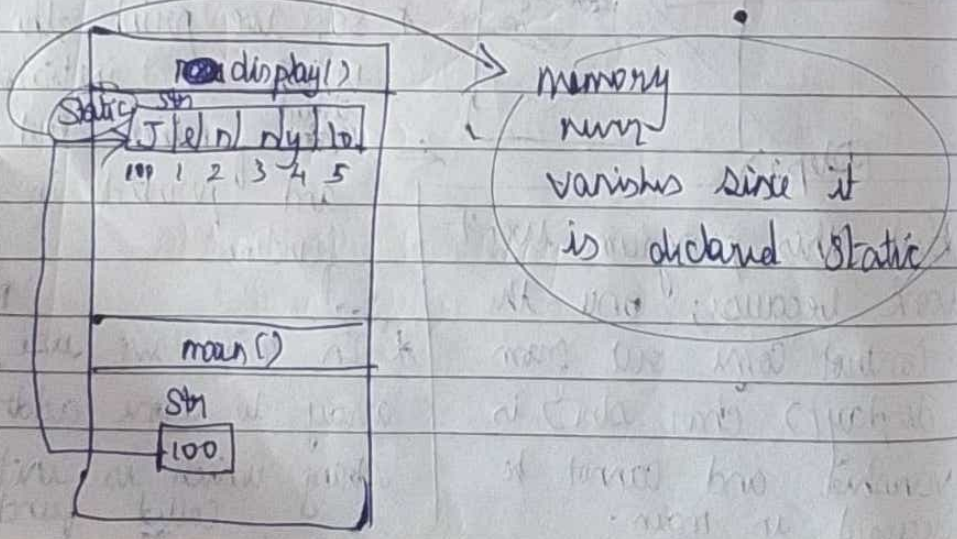
* In main we use character pointer array to store address of string which is initialized in called function.

Solution for example 2:

* Now to overcome this; we initialize string as "static", so that it will be stored in memory of heap (or) static (or) separate location but not in stack memory of display()

Example 3:

```
Char* display();

void main()
{
Char *str;
   str = display();
   printf("string is %s", str);
}
Char* display()
{
Static Char str[] = "Jenny";
   return str;
}
```



Example 4

```
char* display();
void main()
{
    char* str;
    str = display();
    printf("String is %s", str);
}
```

```
char* display()
{
    static char str[] = "Jenny";
    return str;
}
```

```
char *str = "Jenny";
return str;
}
```

Example 4

Explanation

* Because; here str is pointer now, and initializing this pointer with base address of character array / string literal ("Jenny").

* And this pointer variable will have memory in heap area; so the memory not vanished once after out from called function.

* Problem in example 4 -> If we modify the string which is returned to main function; we get run time error because;

we cannot modify the heap memory because heap memory is read only memory.

Example (5)* So how the software developer can overcome this error telling that this is heap memory allocation and value cannot be modified without getting run time error, and to get compile time error when we modify.

Solution :- Example 6.

* We can declare the pointer variable of string as constant which is given is example 5.

Example 5 :-

```
char* display();
void main()
{
    char* str;
    str = display();
    str[0] = 'Z';
    printf("String is %s", str);
}
char* display()
{
    static str[] = "Jenny";
    return str;
}
```

If we modify this str which is static (or) heap memory we get run time error

Example 6:-

```
char * display();
void main()
{
    char *str;
    str = display();
    str[0] = 'z';
    printf("String is %s", str);
}
char * display()
{
    // const char str[] = "Jenny";
    // const char *str = "Jenny";
    return str;
}
```

If we modify we get compile time error because we have initialized str pointer as constant

Note:-

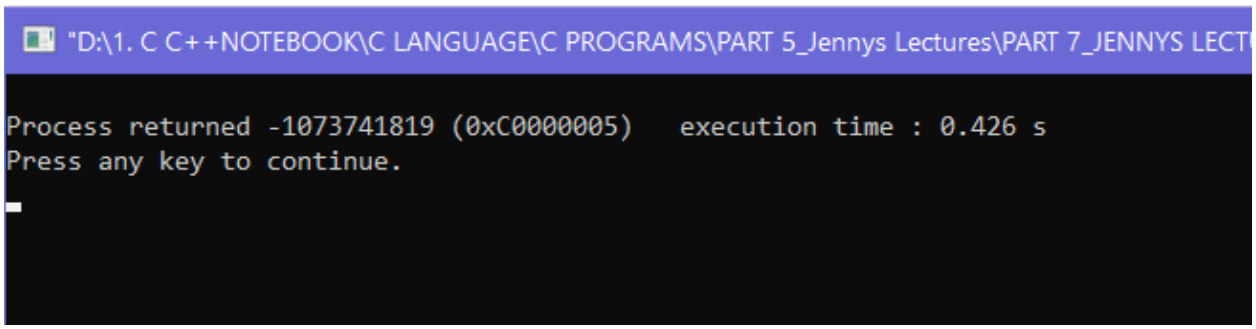
* But if we want to return a string and also to modify it without getting errors;

* Solution is example 3; where we put static and memory is allocated in stack memory and life time of ~~pointer~~ ~~variable~~ character array of str is through out the program..

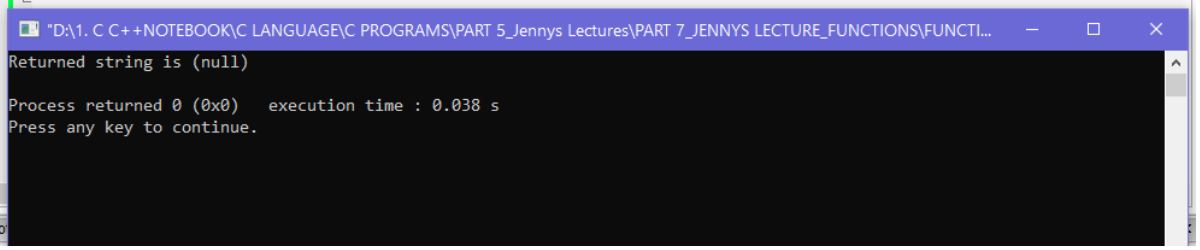
```c
#include <stdio.h>
#include <stdlib.h>
/** 1-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();
printf("Returned string is %s\n",str);
}

char* display()
{
 return "Jayanthi";
 //stored in heap memory, so we cannot modify. It is only read only
}
```

```
"D:\1. C C++NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 7_JENNYS LECTURE_FUNCTIONS\FUNCTI...
Returned string is Jayanthi

Process returned 0 (0x0)   execution time : 0.065 s
Press any key to continue.
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 2-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();
str[0]='j';
//if we modify then in output we get nothing
printf("Returned string is %s\n",str);
}

char* display()
{
 return "Jayanthi";
 //stored in heap memory, so we cannot modify. It is only read only
}
```




```c
#include <stdio.h>
#include <stdlib.h>
/** 3-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();
printf("Returned string is %s\n",str);
}

char* display()
{
 char str[]="Jayanthi";
 return str;
 //string is defined local to this display(), so when control goes out of function
 // its memory is vanished and in main() we cannot display it
}
```



```c
#include <stdio.h>
#include <stdlib.h>
/** 4-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();
str[0]='j';//we can also modify string
printf("Returned string is %s\n",str);
getch();
}

char* display()
{
 static char str[]="Jayanthi";
 return str;
}

//1. when static is given then lifetime of string will through out the program
//2. Even after the control of function display goes out of it, memory for string wont
//get vanishes
```

```
"D:\1. C C++NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 7_JENNYS LECTURE_FUNCTIONS\FUN
Returned string is jayanthi
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 5-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();

printf("Returned string is %s\n",str);
getch();
}

char* display()
{
 char* str="Jayanthi";//another way of declaring string
 return str;
}
```

```
"D:\1. C C++NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 7_JENNYS LECTURE_FUNCTIONS\FUNCTI...
Returned string is Jayanthi
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 6-RETURN STRING FROM FUNCTION **/

char* display();

int main()
{
char* str;
str=display();
printf("Returned string is %s\n",str);
str[0]='j';
//We cannot modify string because in display(), string is initialized in heap memory
printf("Returned string is %s\n",str);
getch();
}

char* display()
{
 char* str="Jayanthi";//another way of declaring string
 //string is initialized in heap memory which is only read only
 //but heap memory wont be vanished, its lifetime is throughout the program
 return str;
}
```

```
"D:\1. C C++NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 7_JENNYS LECTURE_FUNCTIONS\FUNCTI...
Returned string is Jayanthi

Process returned -1073741819 (0xC0000005)   execution time : 0.431 s
Press any key to continue.
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 7-RETURN STRING FROM FUNCTION **/

const char* display();
int main()
{
const char* str;
str=display();
printf("Returned string is %s\n",str);
str[0]='j';
//We cannot modify string because in display(), string is initialized in heap memory
printf("Returned string is %s\n",str);
getch();
}
//We get runtime error and to overcome it by giving it as compile error, we declare as const
const char* display()
{
 char* str="Jayanthi";//another way of declaring string
 //string is initialized in heap memory which is only read only
 //but heap memory wont be vanished, its lifetime is throughout the program
 return str;
```

```
Logs & others
Code::Blocks | Search results | Cccc | Build log | Build messages | CppCheck/Vera++ | CppCheck/Vera++ messages | Cscope | Debug

File            Line   Message
D:\1. C C++...  11     error: assignment of read-only location '*str'
D:\1. C C++...  14     warning: implicit declaration of function 'getch'; did you mean 'getc'? [-W...
                       === Build failed: 1 error(s), 1 warning(s) (0 minute(s), 0 second(s)) ===
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 8-RETURN STRING FROM FUNCTION **/
const char* display();
int main()
{
char* str;
str=display();
printf("Returned string is %s\n",str);
str[0]='j';
//We cannot modify string because in display(), string is initialized in heap memory
printf("Returned string is %s\n",str);
getch();
}
//We get runtime error and to overcome it by giving it as compile error, we declare as const
const char* display()
{
 char* str="Jayanthi";//another way of declaring string
 //string is initialized in heap memory which is only read only
 //but heap memory wont be vanished, its lifetime is throughout the program
 return str;
}
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 8-RETURN STRING FROM FUNCTION **/
const char* display();
int main()
{
char* str; //if we dont specify it as const, then return type of display() const is discarded
str=display();
printf("Returned string is %s\n",str);
str[0]='j';
//We cannot modify string because in display(), string is initialized in heap memory
printf("Returned string is %s\n",str);
getch();
}
//We get runtime error and to overcome it by giving it as compile error, we declare as const
const char* display()
{
 char* str="Jayanthi";//another way of declaring string
 //string is initialized in heap memory which is only read only
 //but heap memory wont be vanished, its lifetime is throughout the program
 return str;
```

Logs & others

Code::Blocks | Search results | Cccc | Build log | Build messages | CppCheck/Vera++ | CppCheck/Vera++ messages | Cscope | Debu

File	Line	Message
		=== Build: Debug in 8_RETURN STRING FROM FUNCTION (compiler: GNU GCC Compil...
D:\1. C C++...		In function 'main':
D:\1. C C++...	8	warning: assignment discards 'const' qualifier from pointer target type [-Wdiscarded-qualifiers]

```
"D:\1. C C++NOTEBOOK\C LANGUAGE\C PROGRAMS\PART 5_Jennys Lectures\PART 7_JENNYS LECTURE_FUNCTIONS\FUN
Returned string is Jayanthi

Process returned -1073741819 (0xC0000005)   execution time : 0.386 s
Press any key to continue.
```

```c
#include <stdio.h>
#include <stdlib.h>
/** 9-RETURN STRING FROM FUNCTION **/
char* display();
int main()
{
const char* str; //specified as const, but return type of display() is not const
str=display();
printf("Returned string is %s\n",str);
str[0]='j';
//We cannot modify string because in display(), string is initialized in heap memory
printf("Returned string is %s\n",str);
getch();
}
//We get runtime error and to overcome it by giving it as compile error, we declare as const
char* display()
{
 char* str="Jayanthi";//another way of declaring string
 //string is initialized in heap memory which is only read only
 //but heap memory wont be vanished, its lifetime is throughout the program
 return str;
```

Logs & others

Code::Blocks | Search results | Cccc | Build log | Build messages | CppCheck/Vera++ | CppCheck/Vera++ messages | Cscope | Debug

File	Line	Message
		=== Build: Debug in 9_RETURN STRING FROM FUNCTION (compiler: GNU GCC Compil...
D:\1. C C++...		In function 'main':
D:\1. C C++...	10	error: assignment of read-only location '*str'
D:\1. C C++...	13	warning: implicit declaration of function 'getch'; did you mean 'getc'? [-W