String

“Array of char(s)”

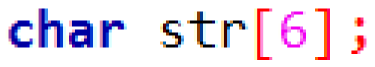
Prerequisite: Array

Find more contents at
https://sites.google.com/view/cse105june18/home

Md. Saidul Hoque Anik
onix.hoque.mist@gmail.com

String

Basically a character array



```
char str[6];
```

String

In place initialization

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

Null character

String

In place initialization

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

Null character

We can also write

```c
char str[6] = "Hello";
```

str:

H	e	l	l	o	\0

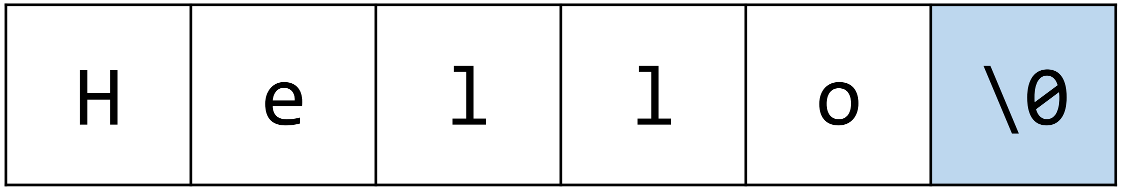
String

Why null character?

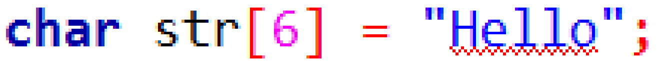
`H`	`e`	`l`	`l`	`o`	`\0`

String

Why null character?



To mark the end of string

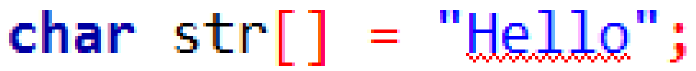


```
char str[6] = "Hello";
```

[C compiler automatically adds null character here]

String

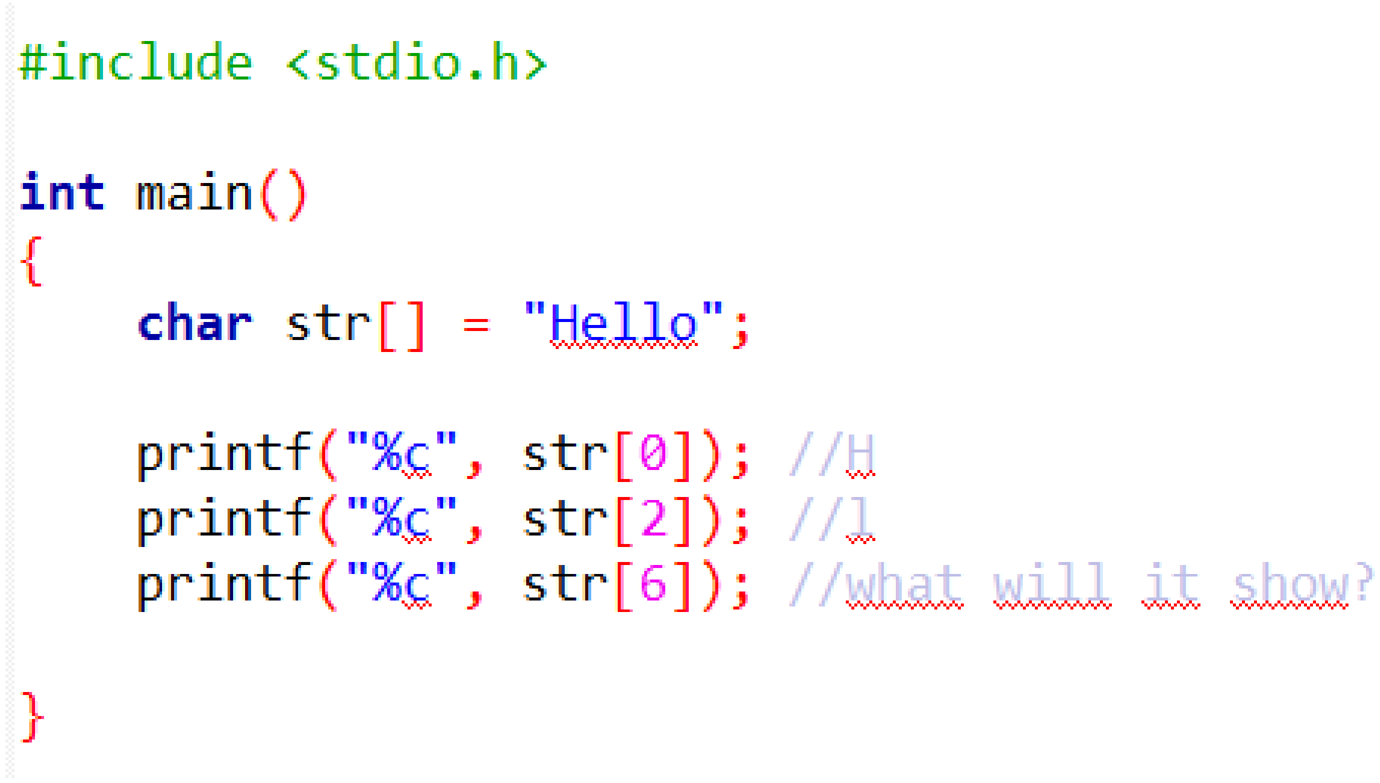
Another more useful way of initialization (without length)



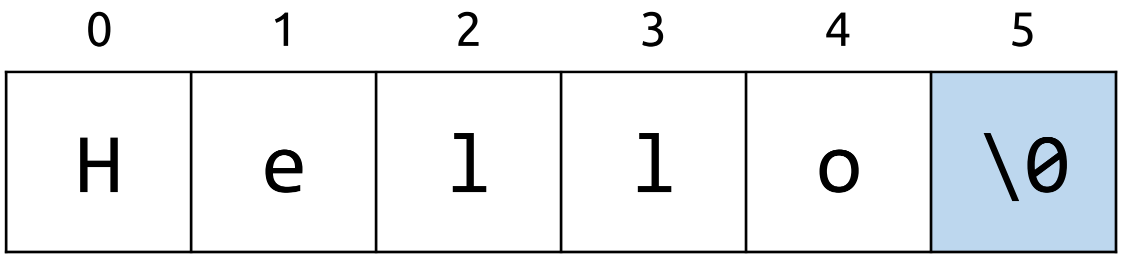
```
char str[] = "Hello";
```

[C compiler automatically assigns required size]

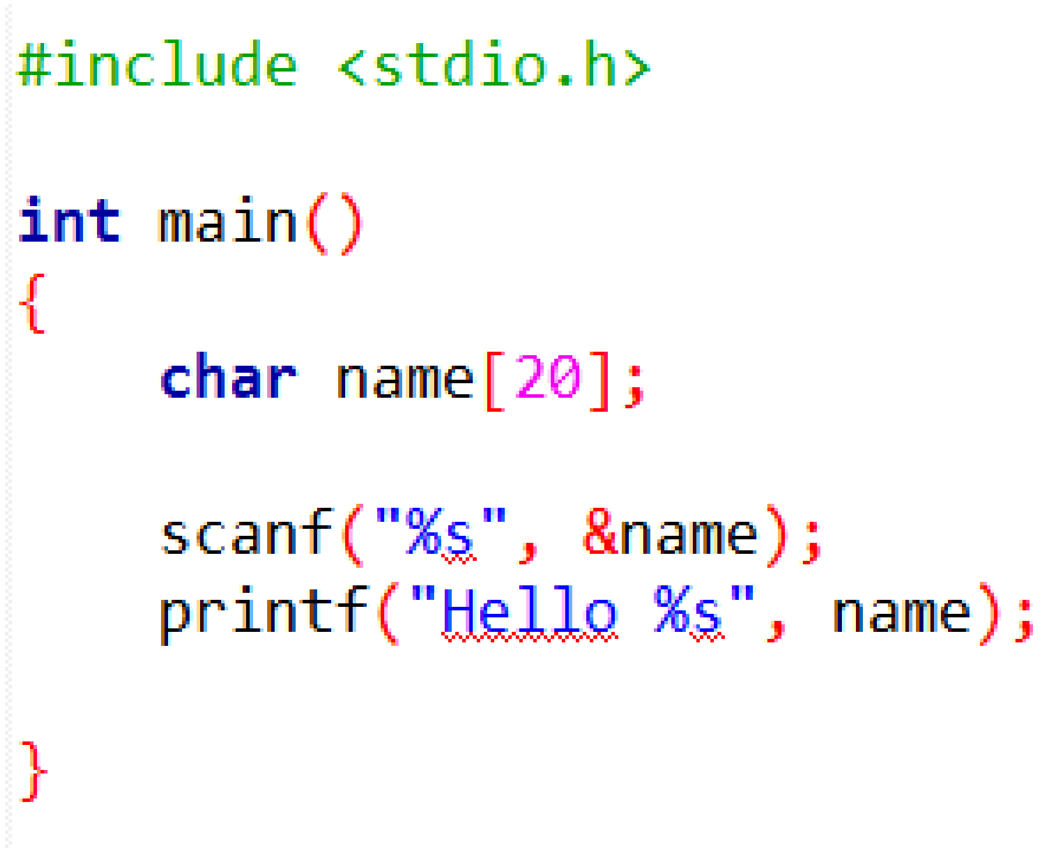
Accessing char by char



```c
#include <stdio.h>

int main()
{
    char str[] = "Hello";

    printf("%c", str[0]); //H
    printf("%c", str[2]); //l
    printf("%c", str[6]); //what will it show?

}
```

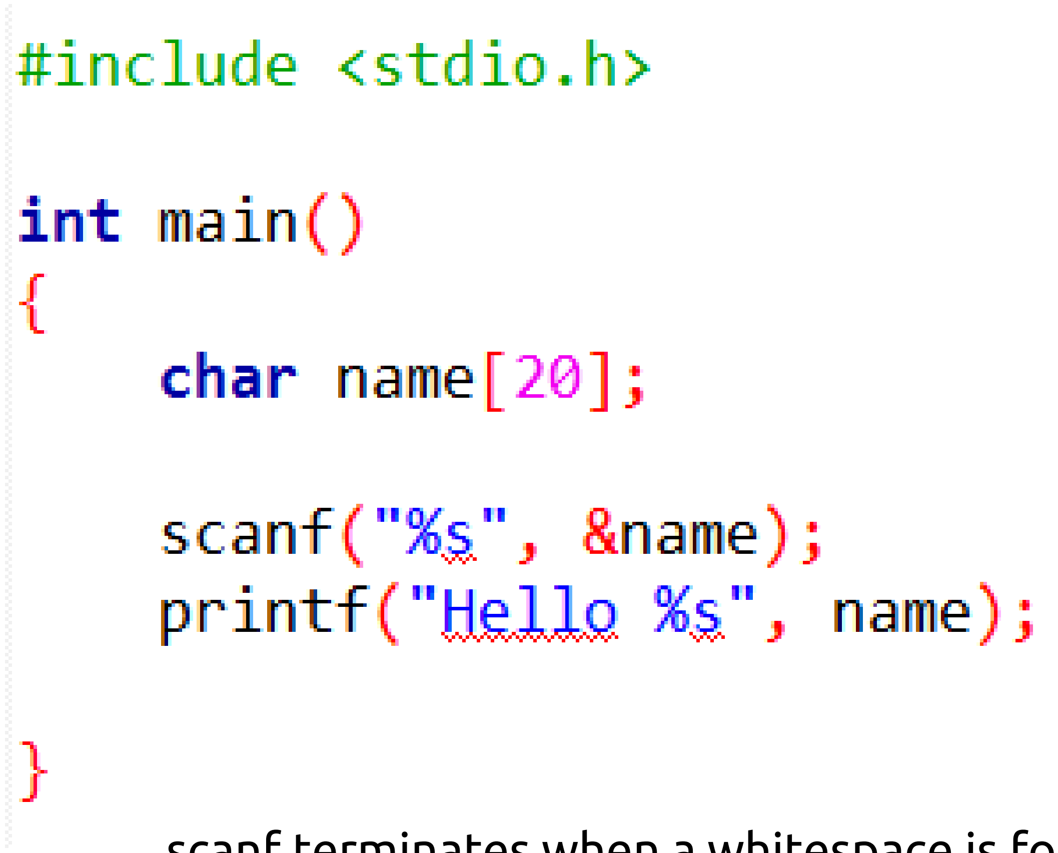


Reading word from user



```c
#include <stdio.h>

int main()
{
    char name[20];

    scanf("%s", &name);
    printf("Hello %s", name);

}
```

Reading word from user



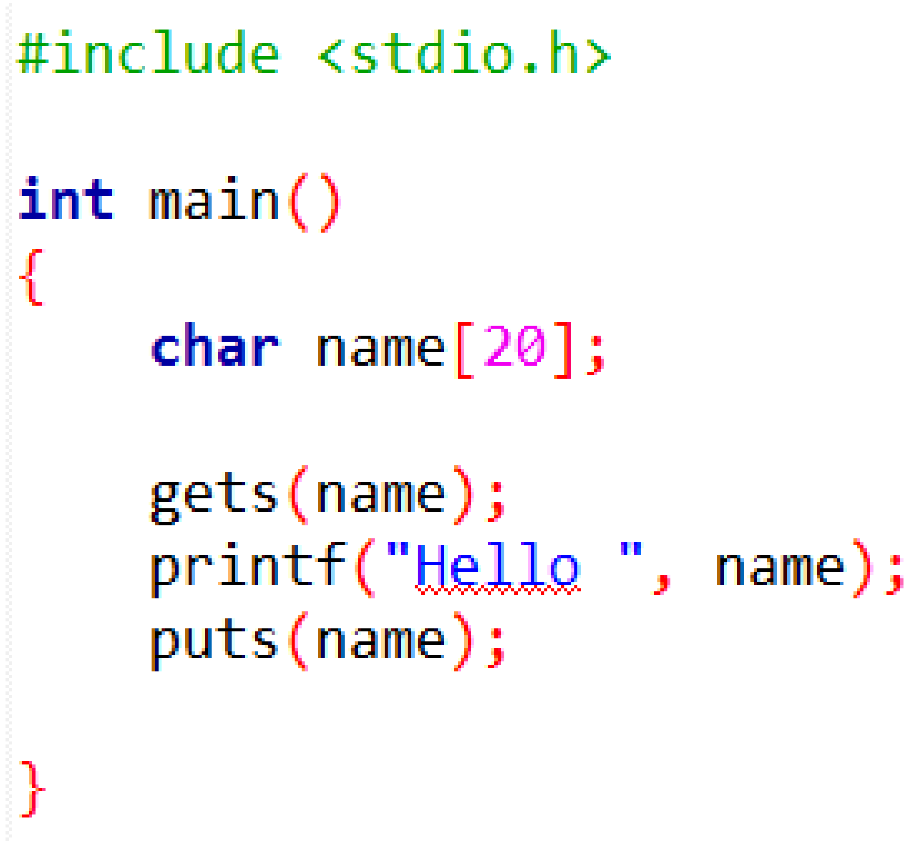
```c
#include <stdio.h>

int main()
{
    char name[20];

    scanf("%s", &name);
    printf("Hello %s", name);

}
```

scanf terminates when a whitespace is found.

So we cannot use %s for reading a line

Reading/writing line

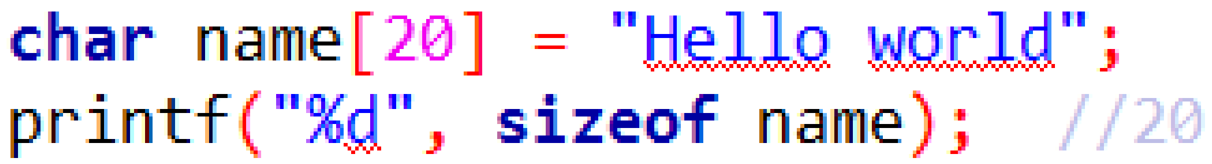
Usage of gets() and puts()



```
#include <stdio.h>

int main()
{
    char name[20];

    gets(name);
    printf("Hello ", name);
    puts(name);

}
```

Finding the length of the string

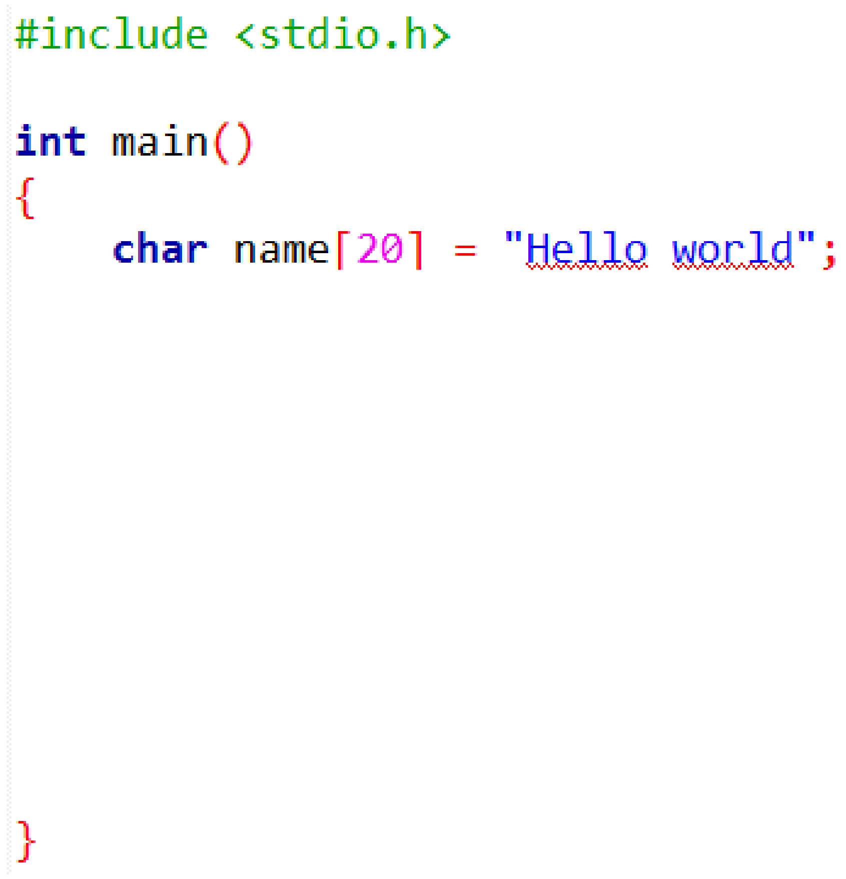
Is it 20?



```
char name[20] = "Hello world";
printf("%d", sizeof name);  //20
```

Finding the length of the string

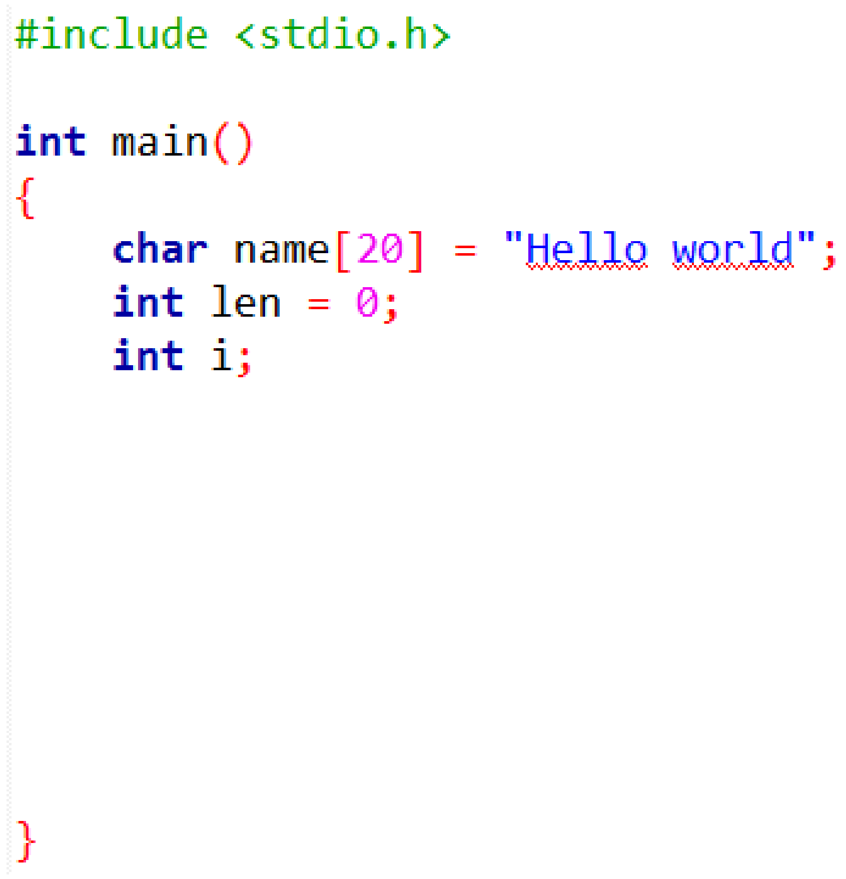
How do we know the end of string?



```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";









}
```

Finding the length of the string

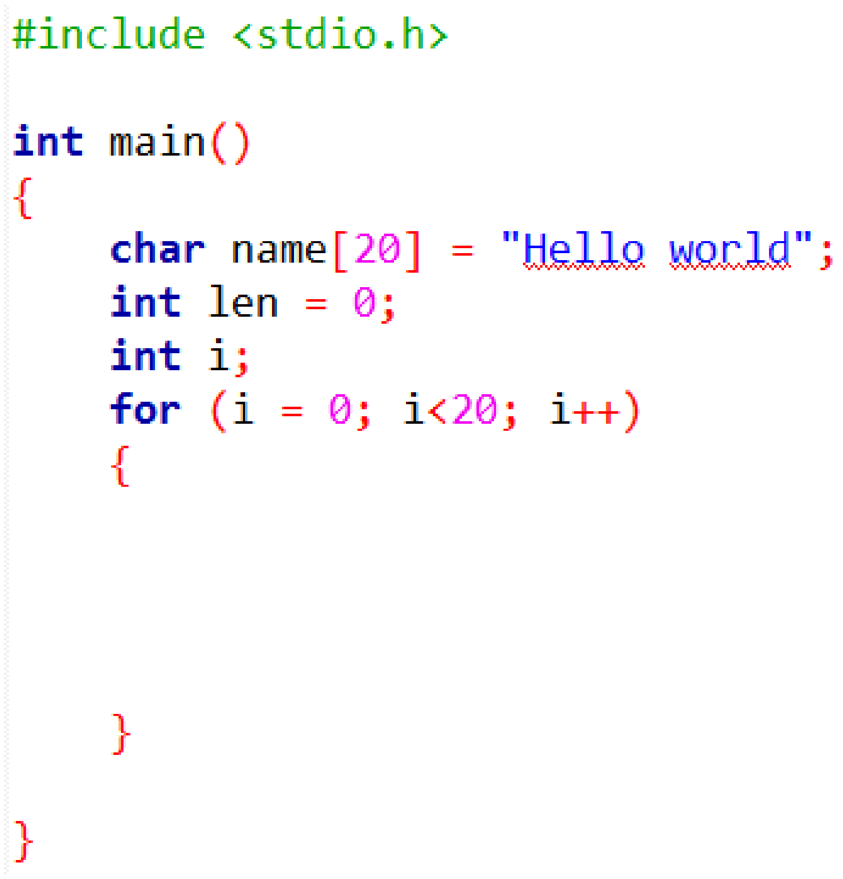
How do we know the end of string?



```
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;









}
```

Finding the length of the string

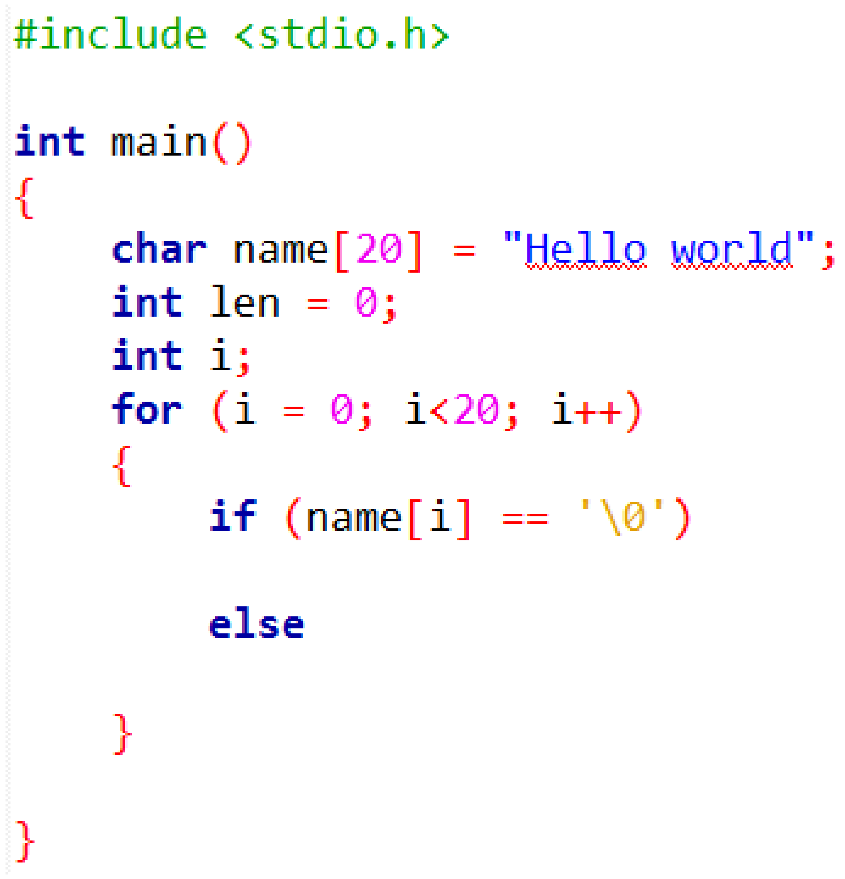
Why 20?



```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;
    for (i = 0; i<20; i++)
    {




    }

}
```

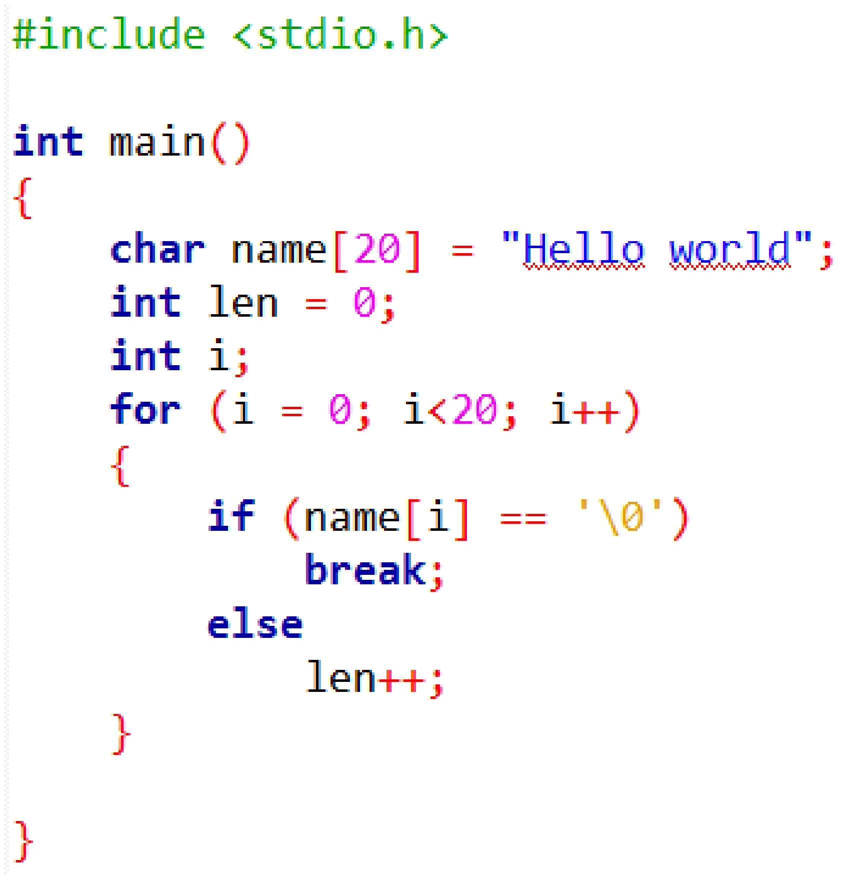
Finding the length of the string

What will happen if null char is found? Else?



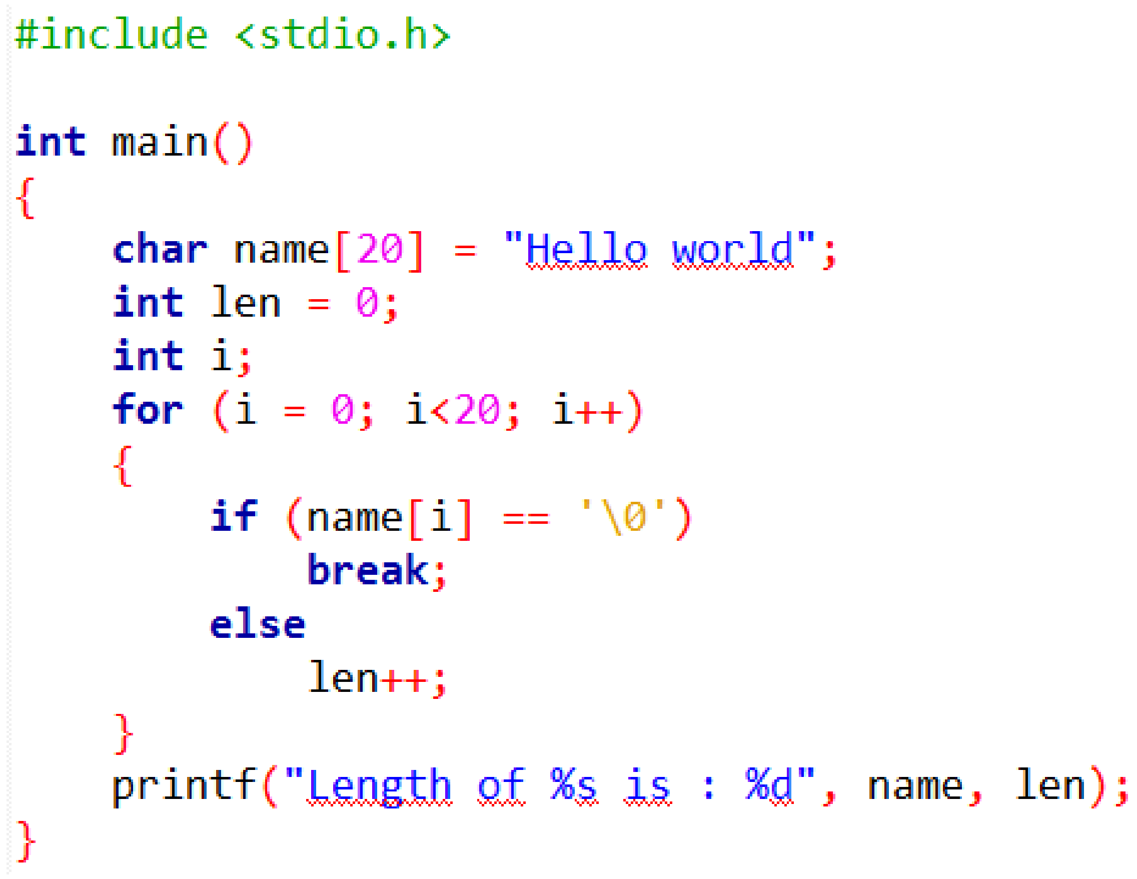
```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;
    for (i = 0; i<20; i++)
    {
        if (name[i] == '\0')

        else

    }

}
```

Finding the length of the string



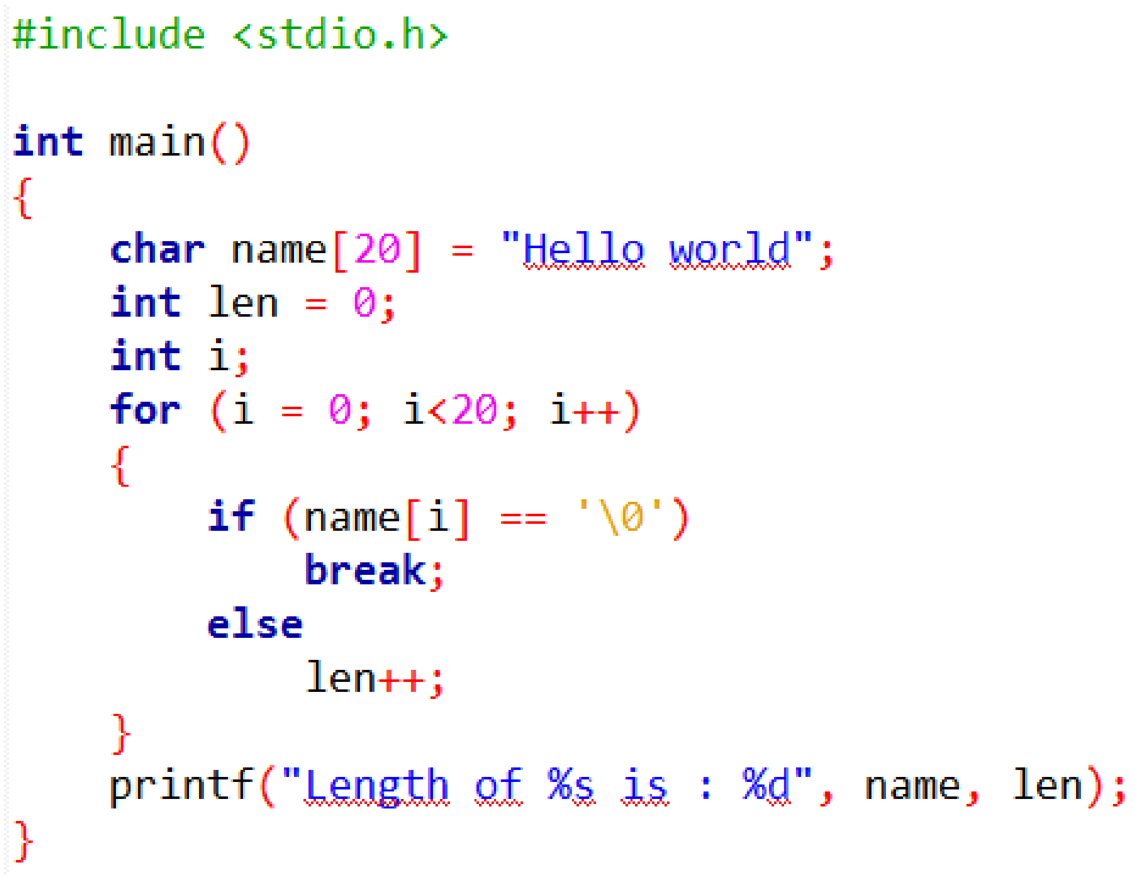
```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;
    for (i = 0; i<20; i++)
    {
        if (name[i] == '\0')
            break;
        else
            len++;
    }

}
```

Finding the length of the string



```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;
    for (i = 0; i<20; i++)
    {
        if (name[i] == '\0')
            break;
        else
            len++;
    }
    printf("Length of %s is : %d", name, len);
}
```

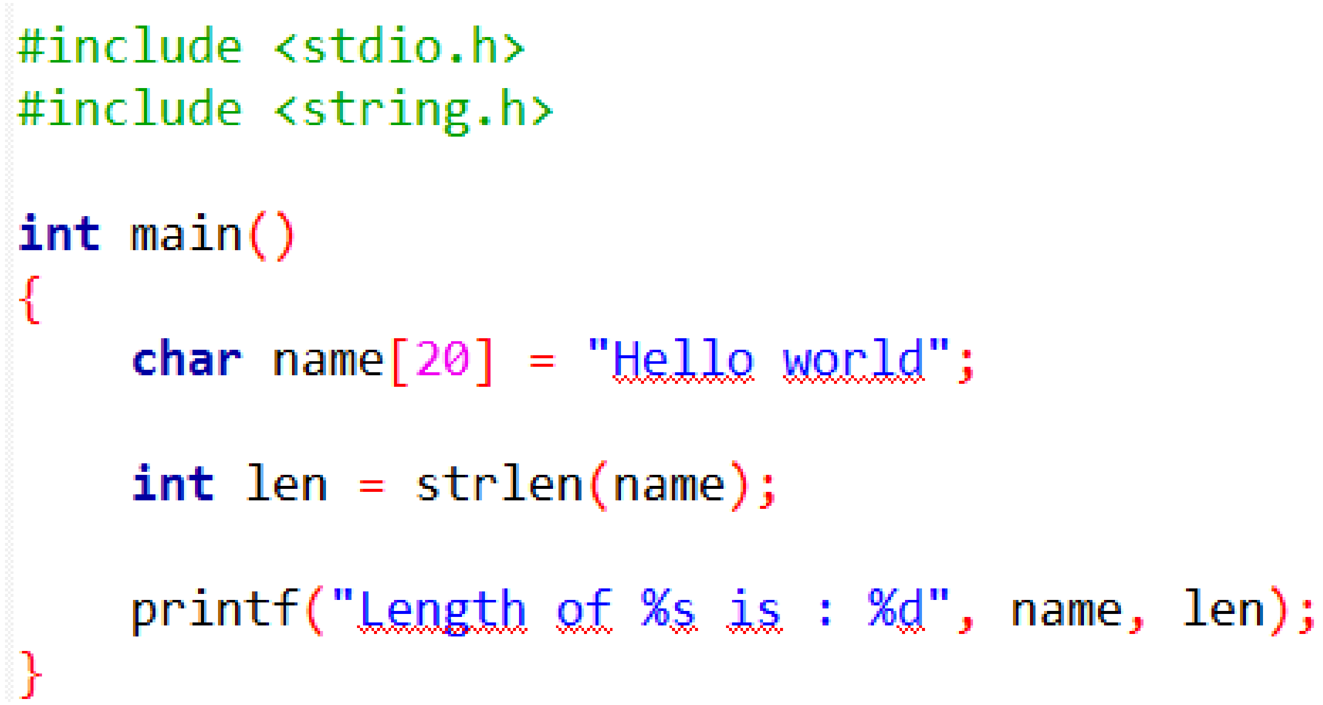
Task 0: Rewrite the following code using while loop



```c
#include <stdio.h>

int main()
{
    char name[20] = "Hello world";
    int len = 0;
    int i;
    for (i = 0; i<20; i++)
    {
        if (name[i] == '\0')
            break;
        else
            len++;
    }
    printf("Length of %s is : %d", name, len);
}
```

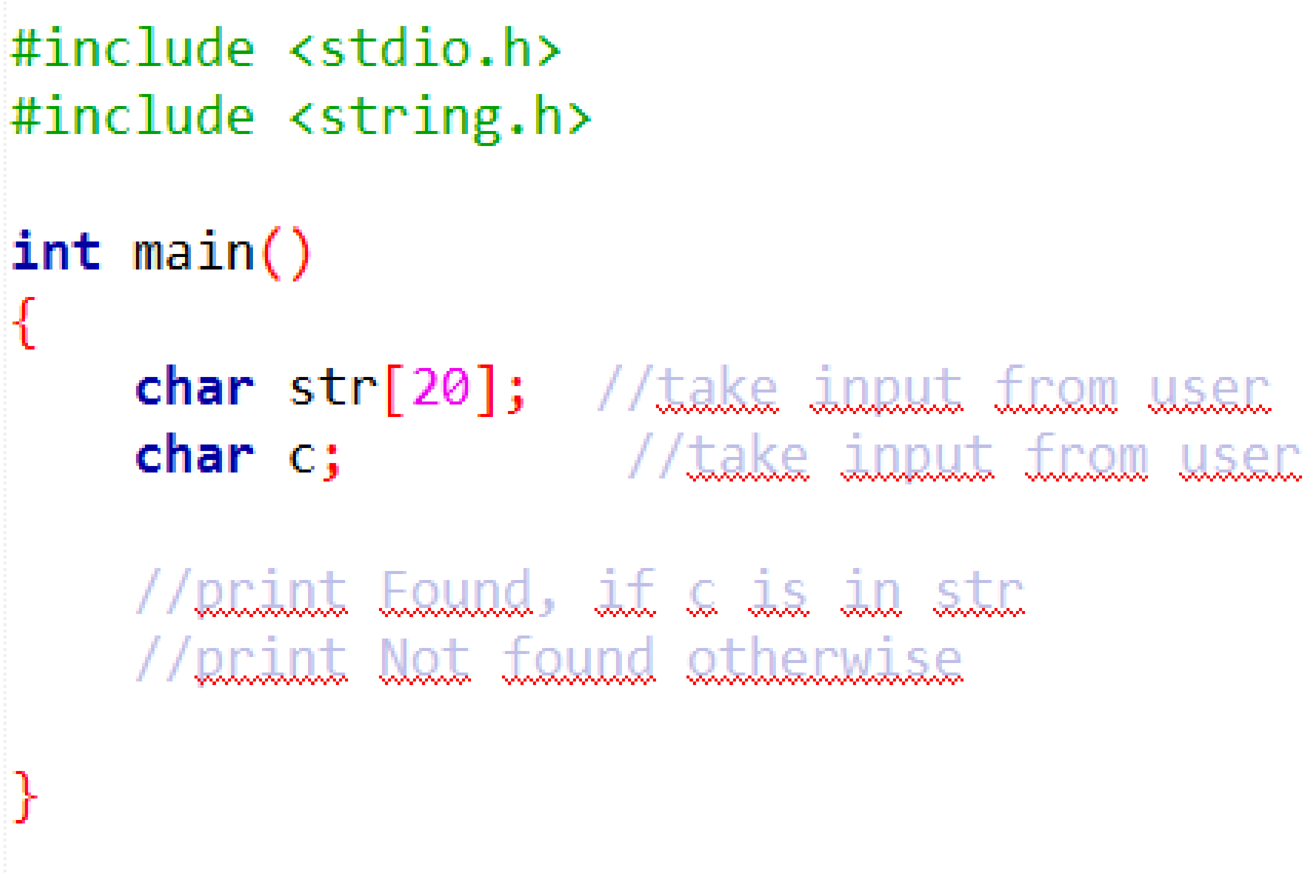
Finding the length of the string

We can also use the library function



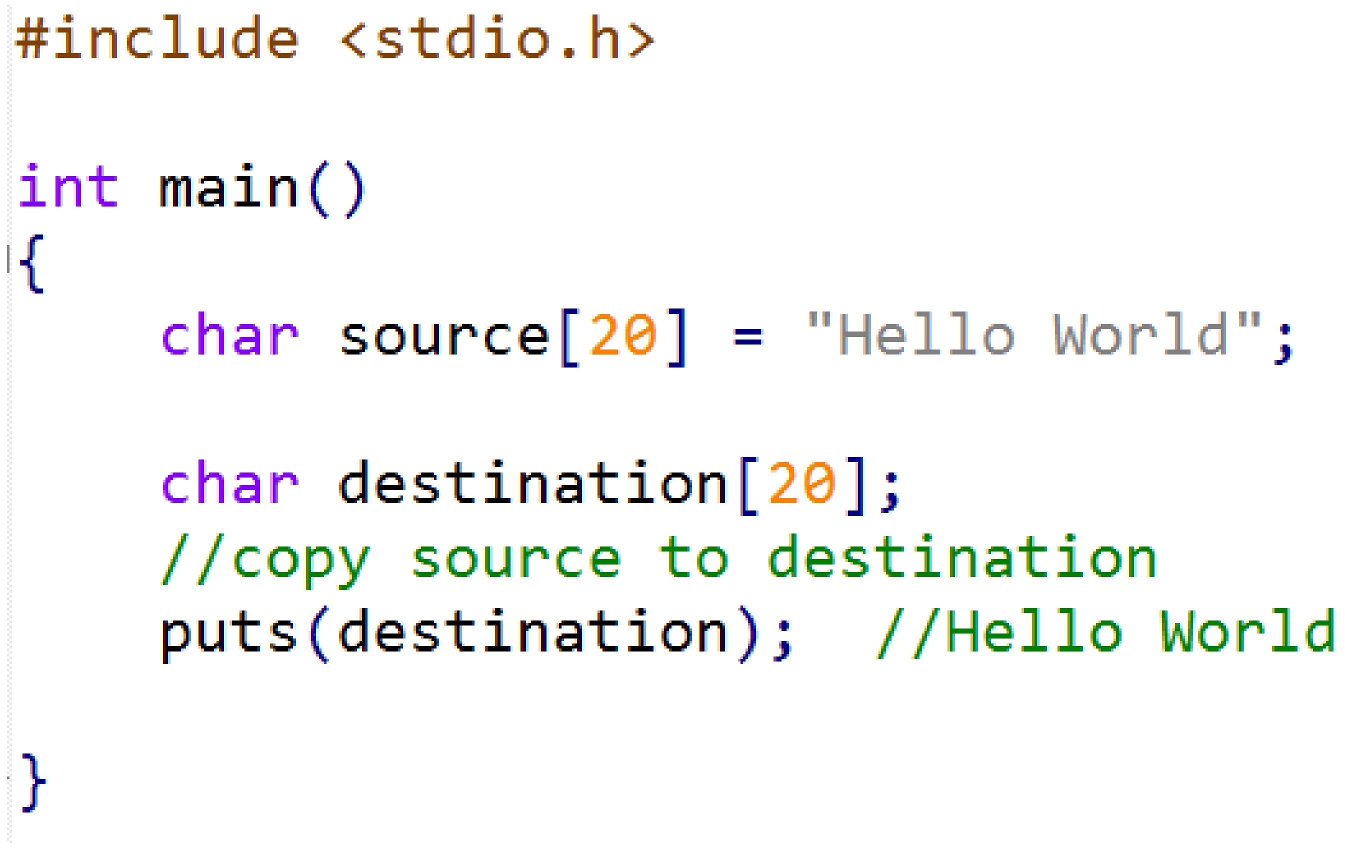
```
#include <stdio.h>
#include <string.h>

int main()
{
    char name[20] = "Hello world";

    int len = strlen(name);

    printf("Length of %s is : %d", name, len);
}
```

Task 1: Search a character in a string



```c
#include <stdio.h>
#include <string.h>

int main()
{
    char str[20];  //take input from user
    char c;         //take input from user

    //print Found, if c is in str
    //print Not found otherwise

}
```

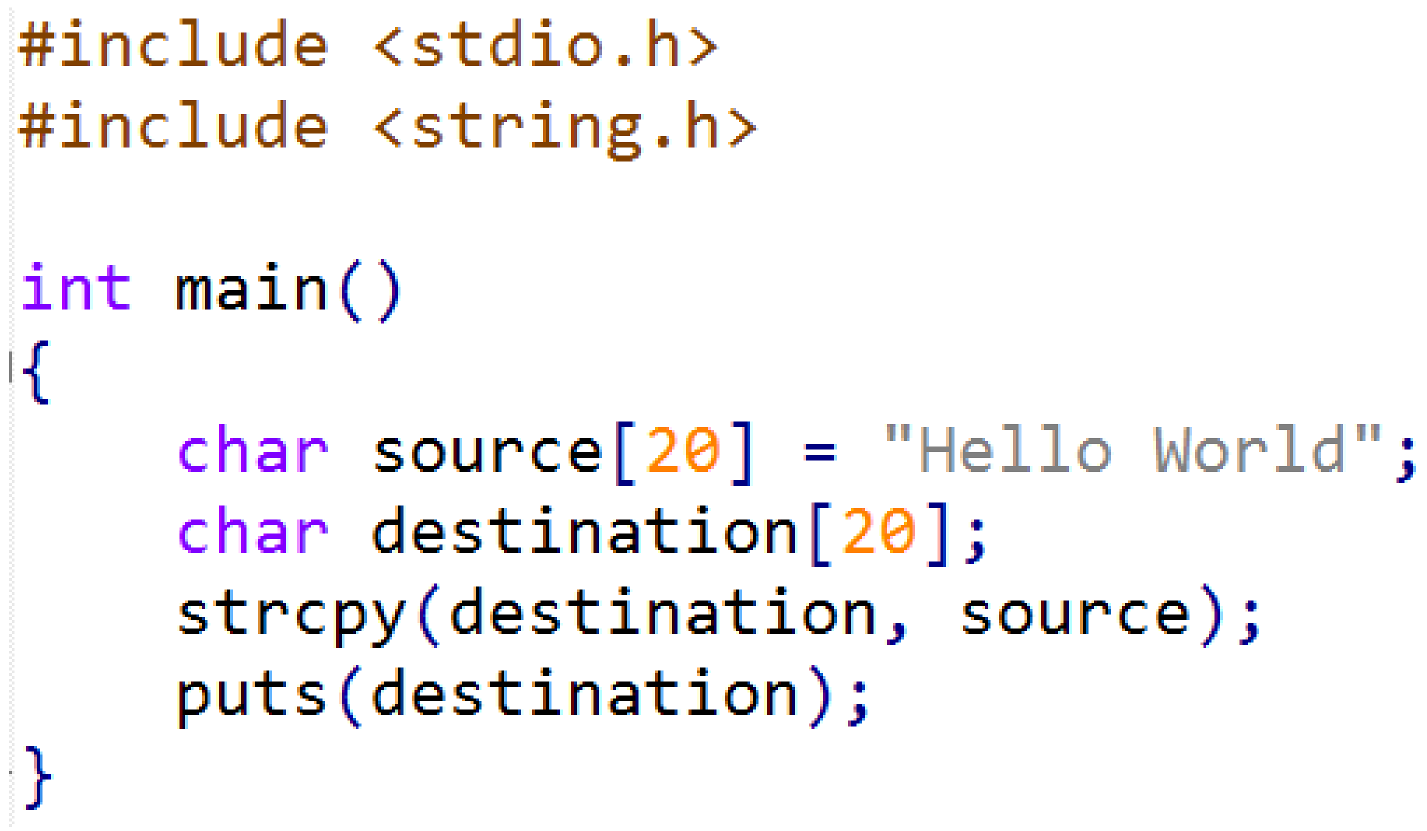
Task 2: Copying one String to another



```c
#include <stdio.h>

int main()
{
    char source[20] = "Hello World";

    char destination[20];
    //copy source to destination
    puts(destination);  //Hello World

}
```

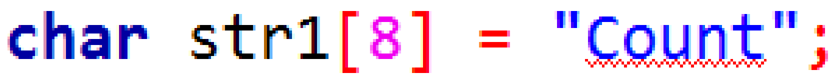
Task 2: Copying one String to another

We can also use the library function



```c
#include <stdio.h>
#include <string.h>

int main()
{
    char source[20] = "Hello World";
    char destination[20];
    strcpy(destination, source);
    puts(destination);
}
```

Task 3: Concatenation (joining)



```
char str1[8] = "Count";
```

Task 3: Concatenation (joining)

```c
char str1[8] = "Count";
```

0	1	2	3	4	5	6	7
C	o	u	n	t	\0		

Task 3: Concatenation (joining)

```
char str1[8] = "Count";
```

0	1	2	3	4	5	6	7
C	o	u	n	t	\0		

```
char str2[5] = "ry";
```

Task 3: Concatenation (joining)

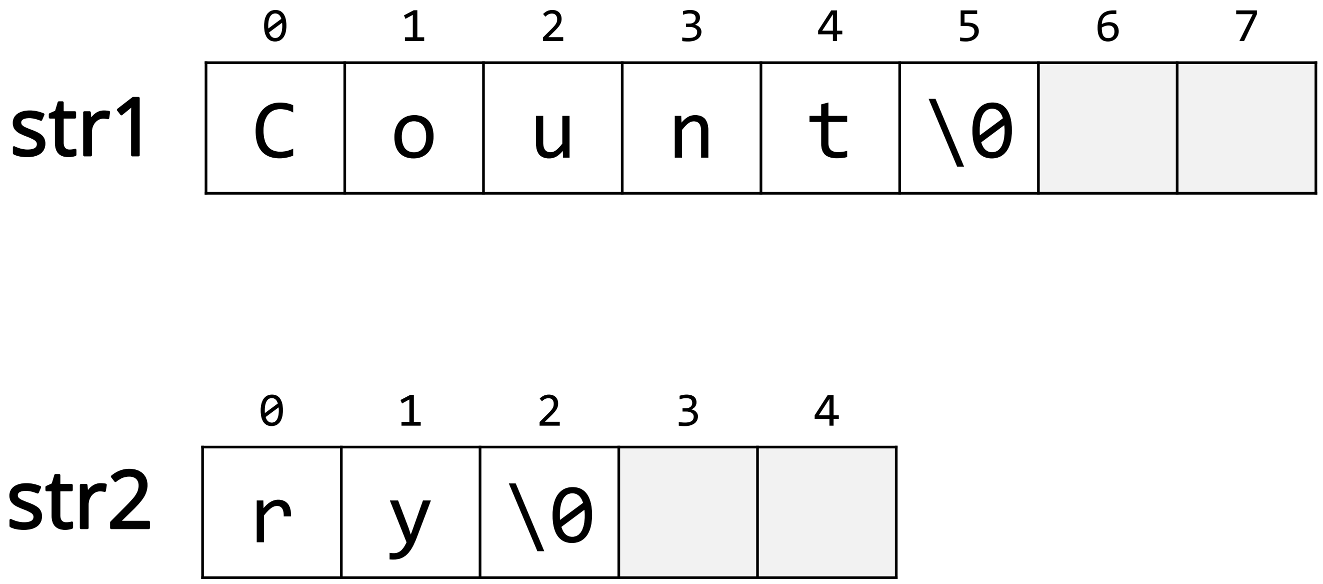
```
char str1[8] = "Count";
```

0	1	2	3	4	5	6	7
C	o	u	n	t	\0		

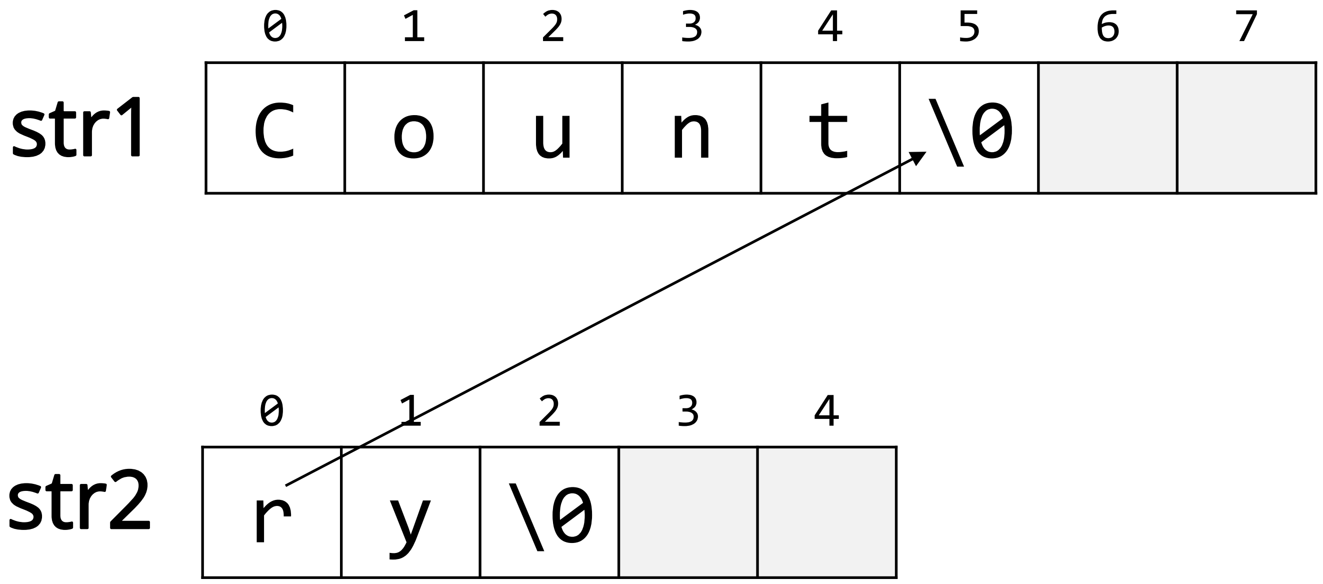
```
char str2[5] = "ry";
```

0	1	2	3	4
r	y	\0		

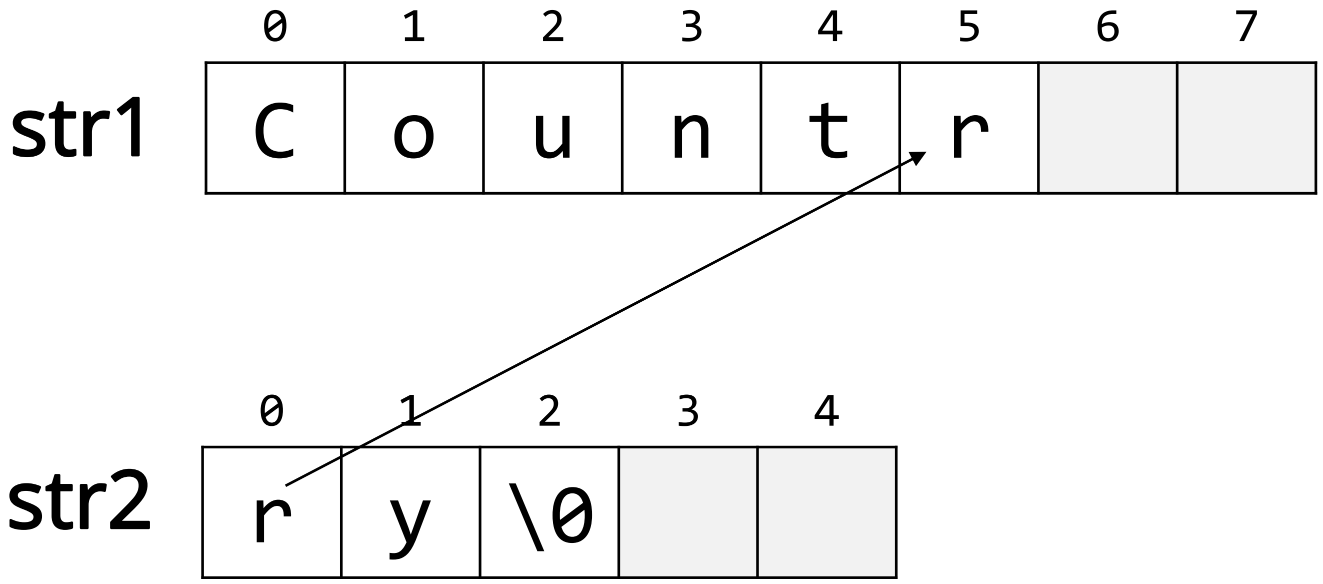
Task 3: Concatenation (joining)



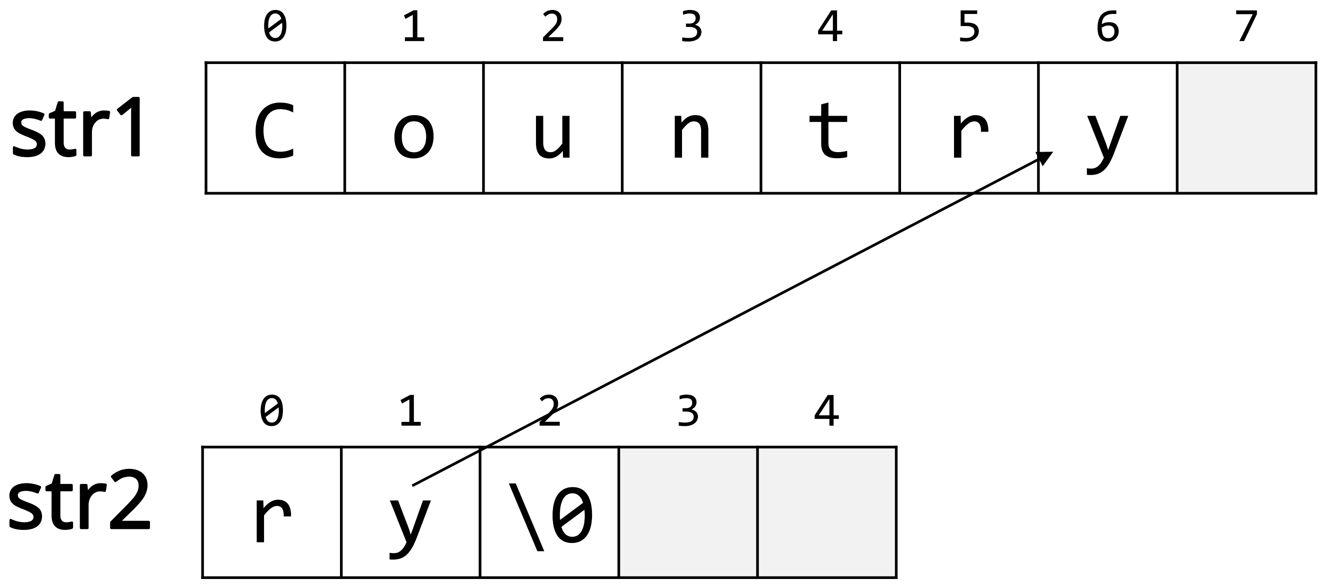
Task 3: Concatenation (joining)



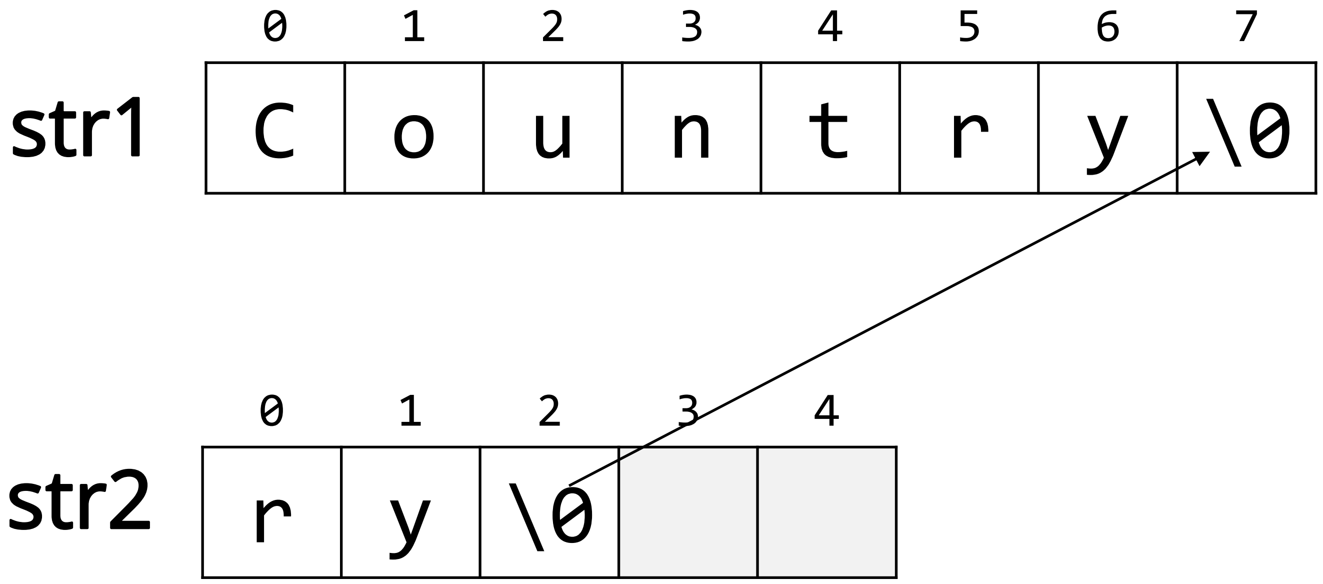
Task 3: Concatenation (joining)



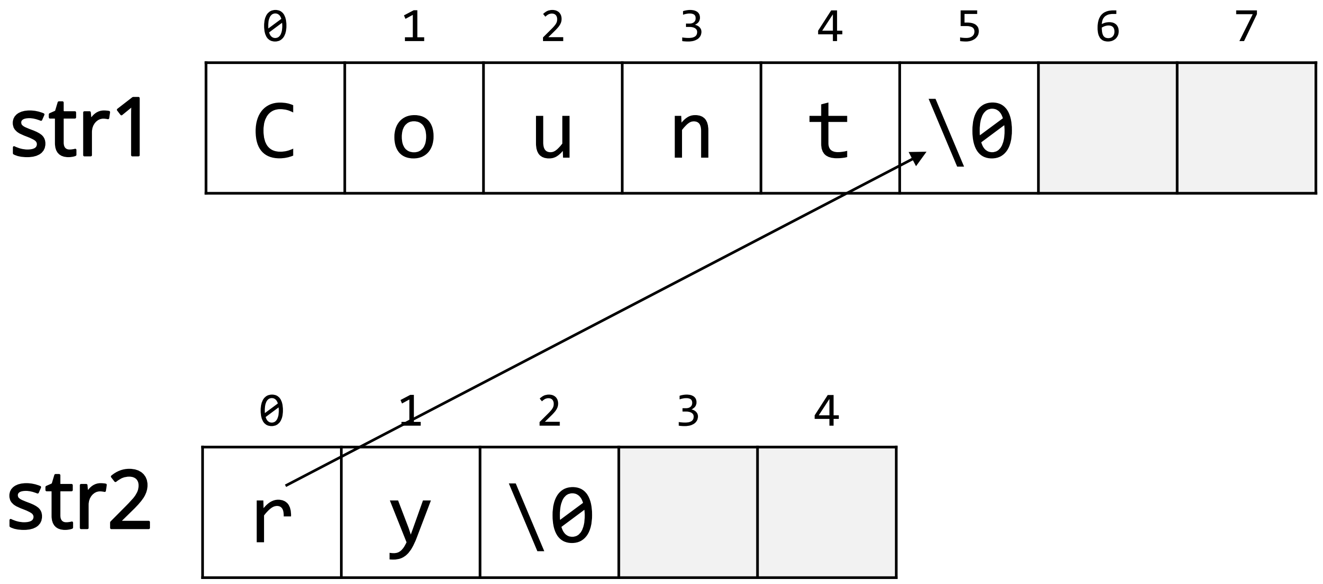
Task 3: Concatenation (joining)



Task 3: Concatenation (joining)

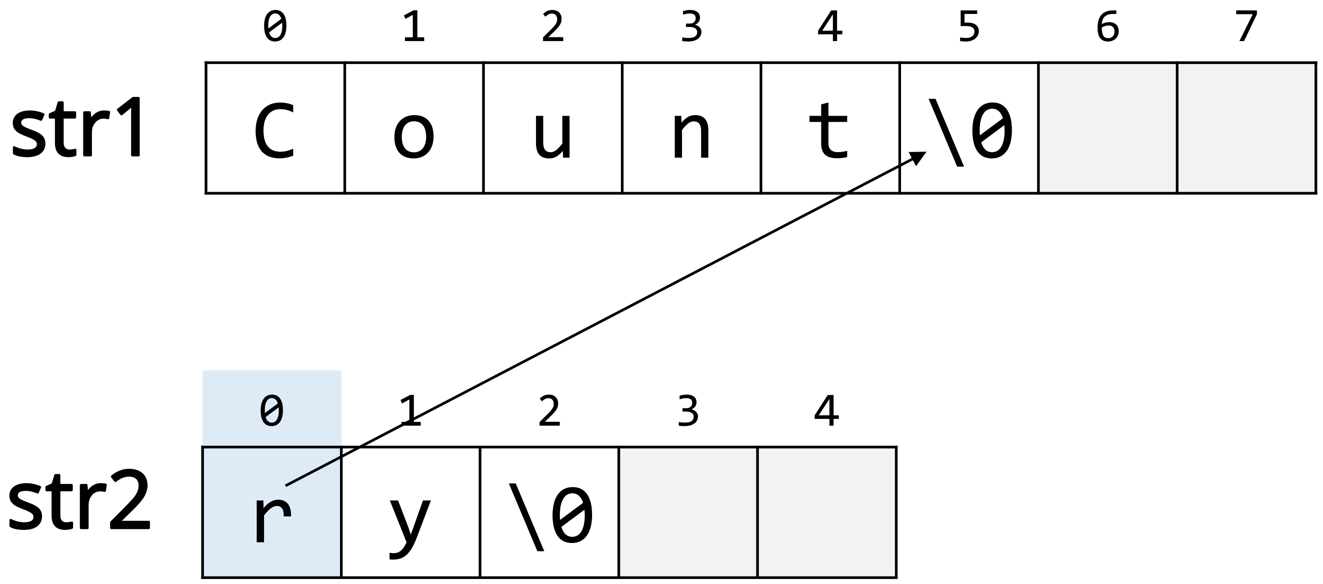


Task 3: Concatenation (joining)



What will be the starting value of i?

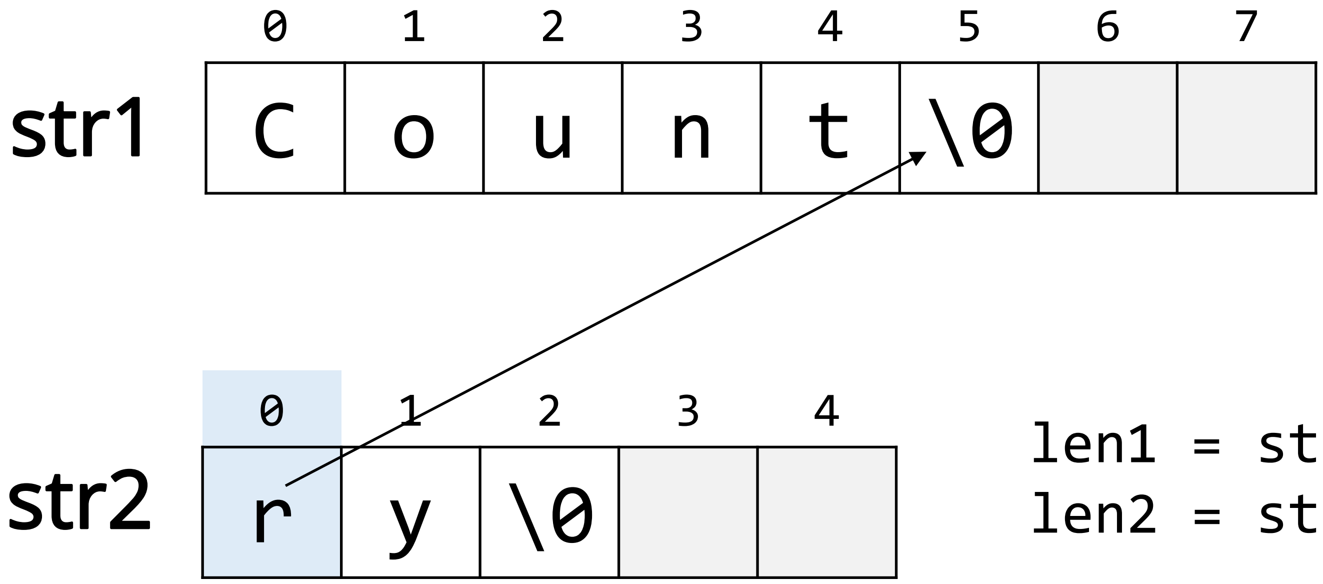
Task 3: Concatenation (joining)



What will be the starting value of i?

```
for (i = 0;
```

Task 3: Concatenation (joining)

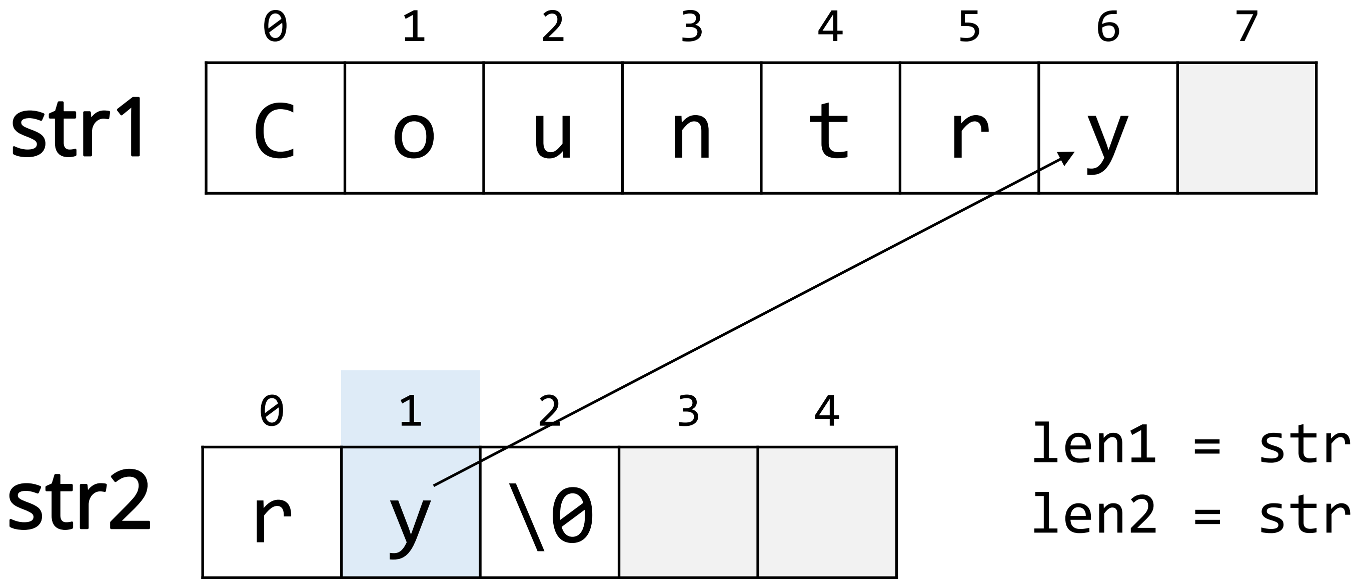


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What will be the ending value of i?

```
for (i = 0;
```

Task 3: Concatenation (joining)

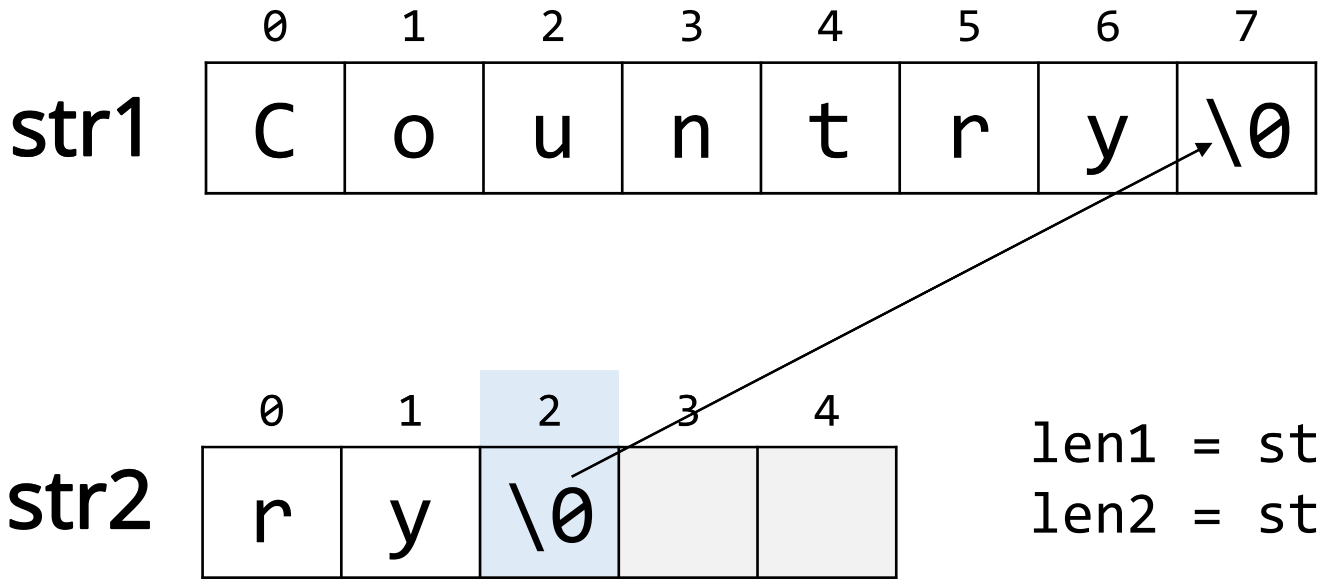


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What will be the ending value of i?

```
    for (i = 0;
```

Task 3: Concatenation (joining)

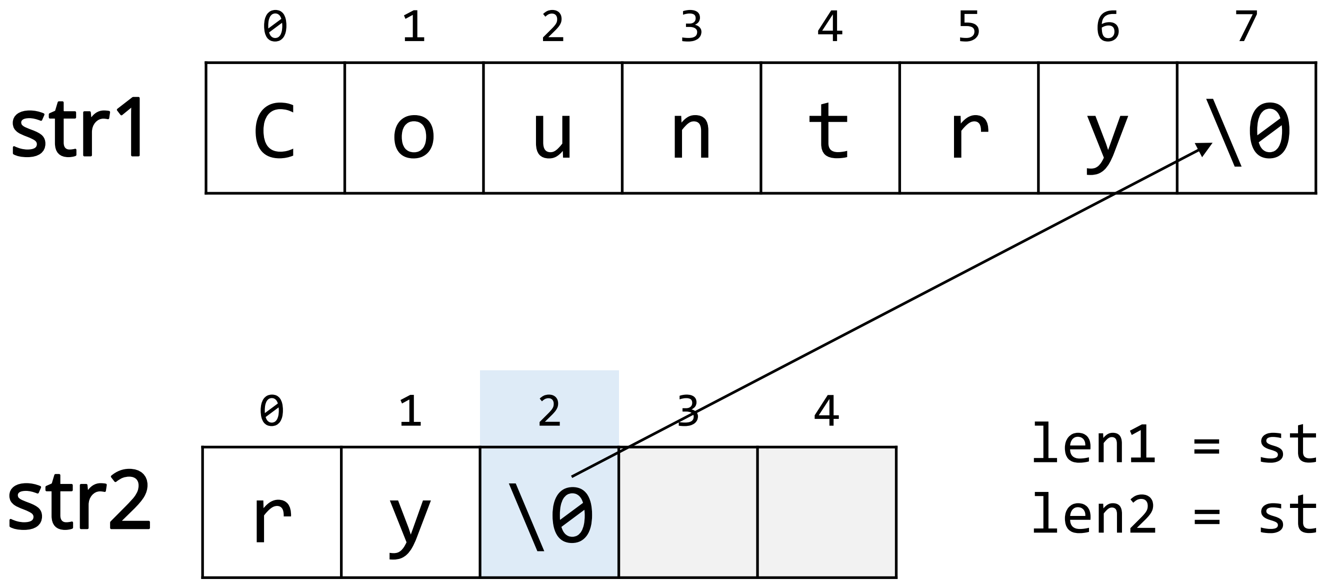


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What will be the ending value of i?

```
for (i = 0;
```

Task 3: Concatenation (joining)

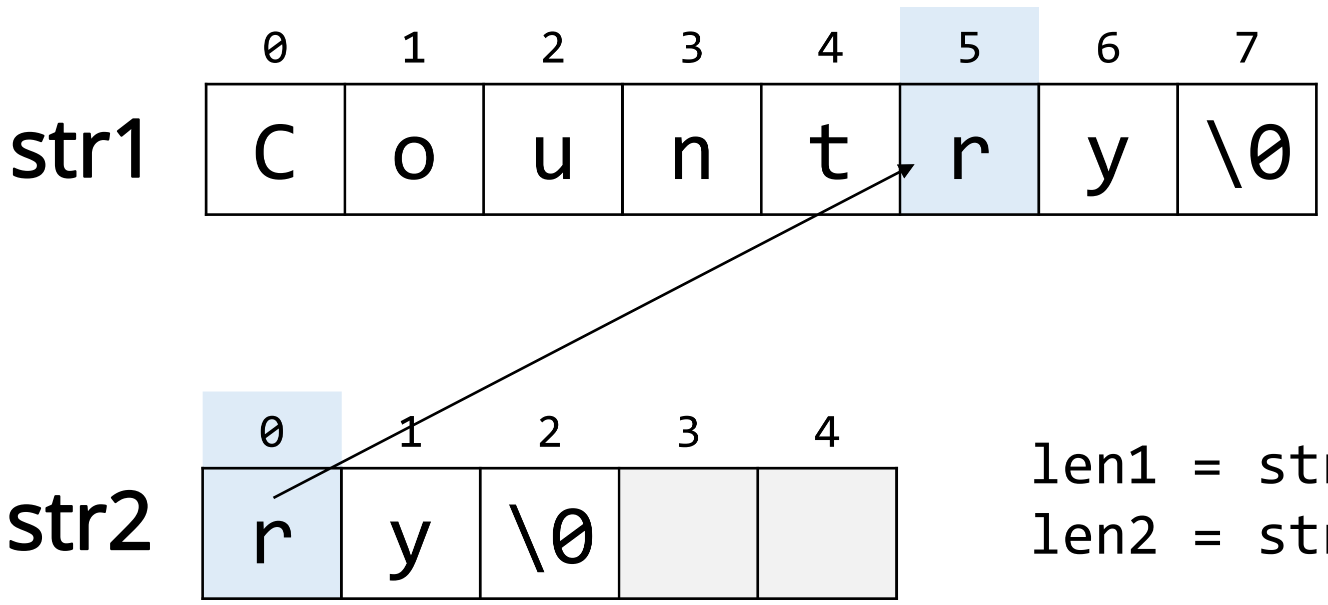


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What will be the ending value of i?

```
for (i = 0; i<=len2; i++)
```

Task 3: Concatenation (joining)

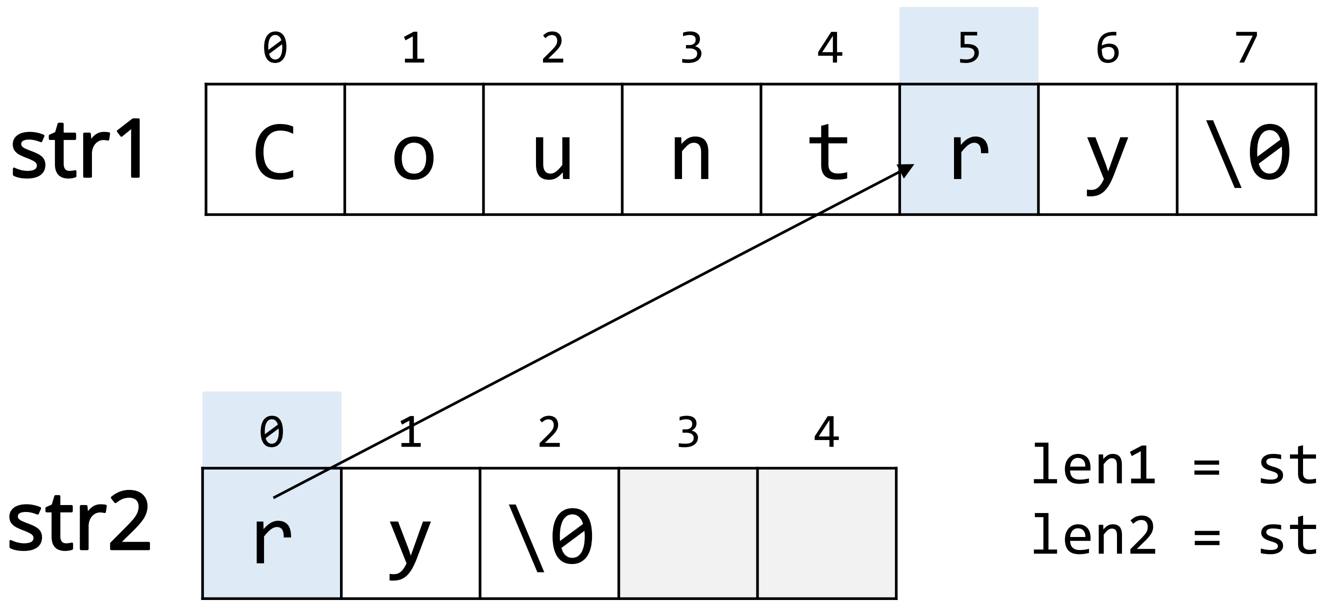


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What is happening in each iteration?

```
for (i = 0; i<=len2; i++)
{     str1[ ? ] = str2[ ? ];  }
```

Task 3: Concatenation (joining)

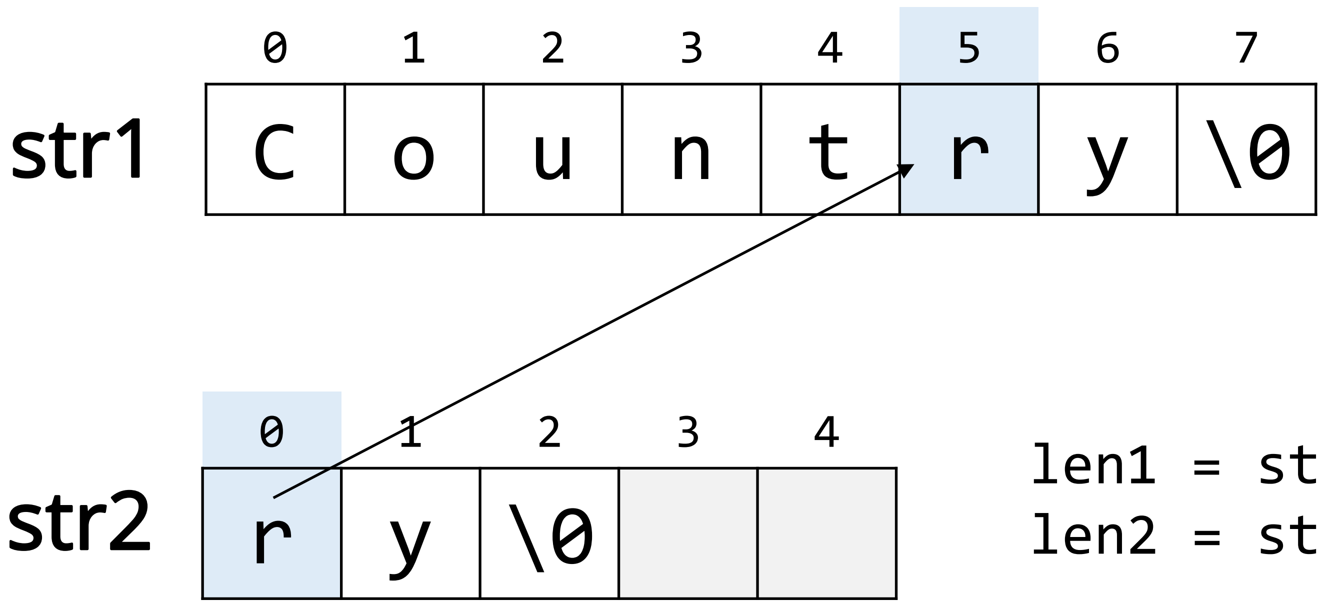


```
len1 = strlen(str1);
len2 = strlen(str2);
```

What is happening in each iteration?

```
for (i = 0; i<=len2; i++)
{     str1[ ? ] = str2[ i ];  }
```

Task 3: Concatenation (joining)



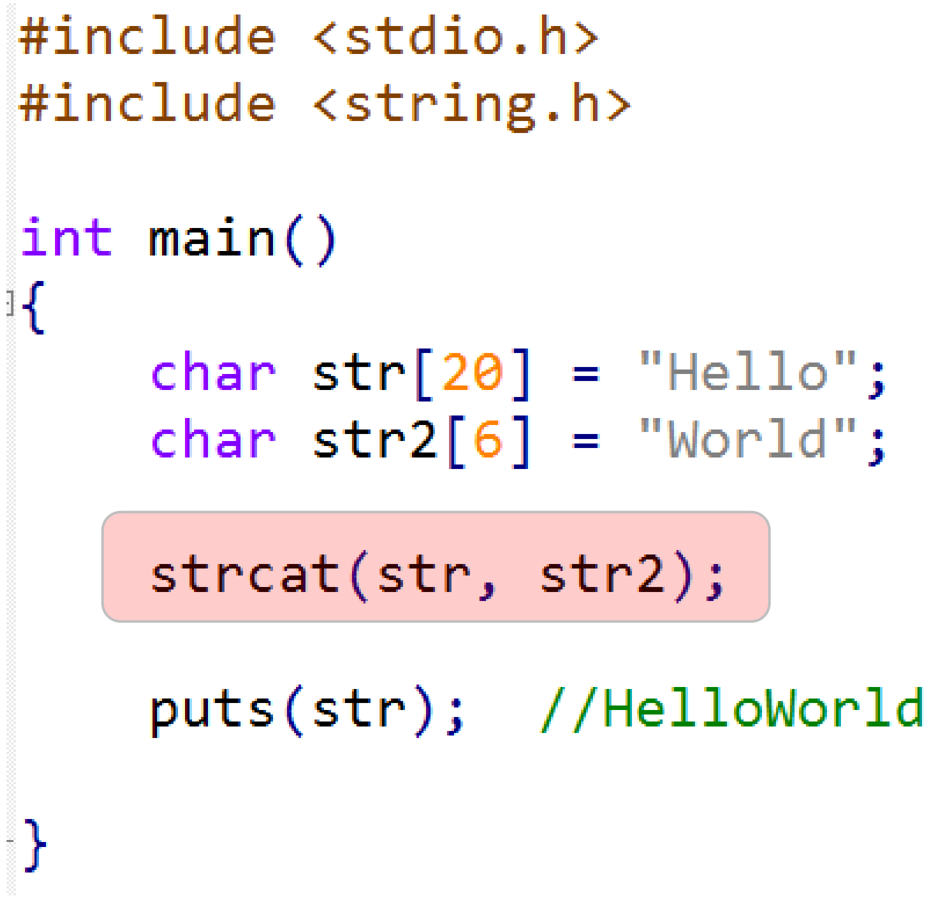
```
len1 = strlen(str1);
len2 = strlen(str2);
```

What is happening in each iteration?

```
for (i = 0; i<=len2; i++)
{     str1[ i + len1 ] = str2[ i ]; }
```

Task 3: Concatenation (joining)

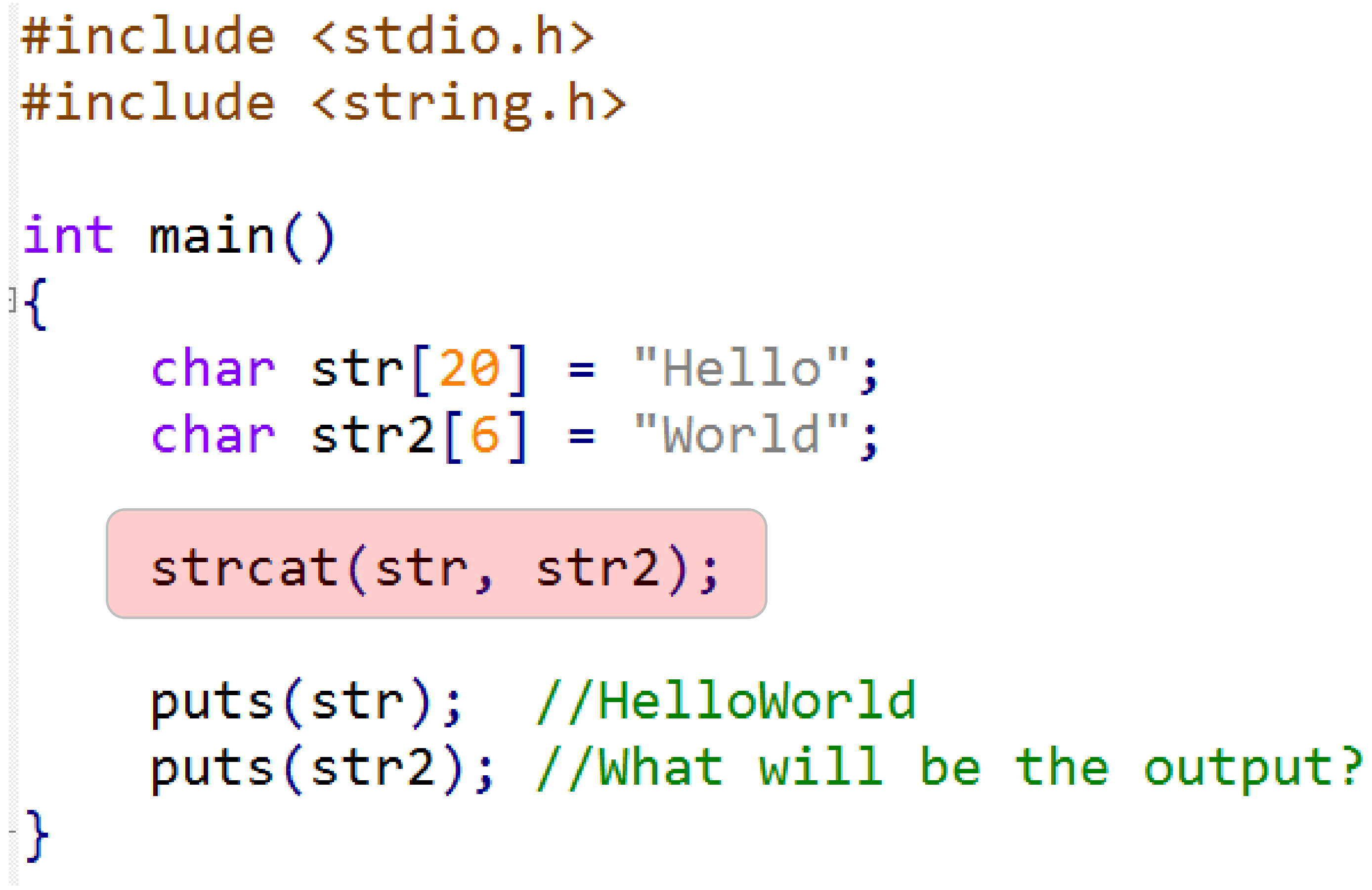
We can also use the library function



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

int main()
{
    char str[20] = "Hello";
    char str2[6] = "World";

    strcat(str, str2);

    puts(str);  //HelloWorld

}
```

Task 3: Concatenation (joining)

We can also use the library function

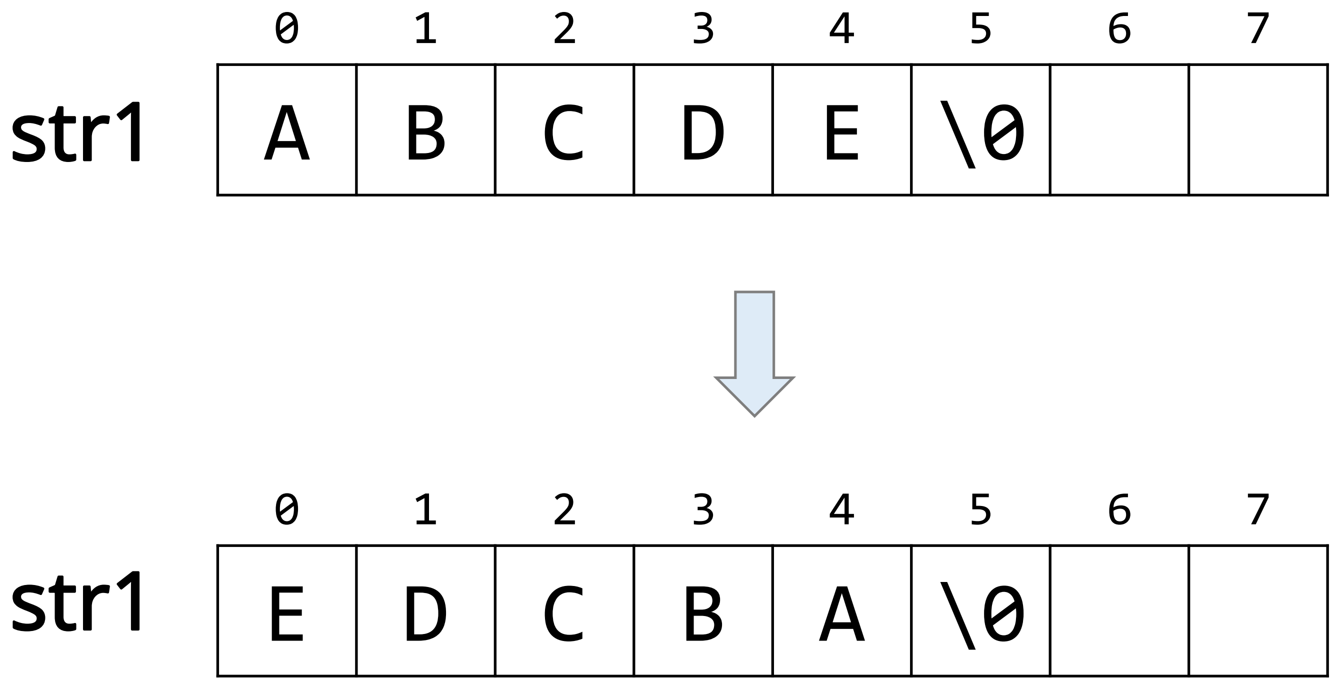


```c
#include <stdio.h>
#include <string.h>

int main()
{
    char str[20] = "Hello";
    char str2[6] = "World";

    strcat(str, str2);

    puts(str);  //HelloWorld
    puts(str2); //What will be the output?
}
```

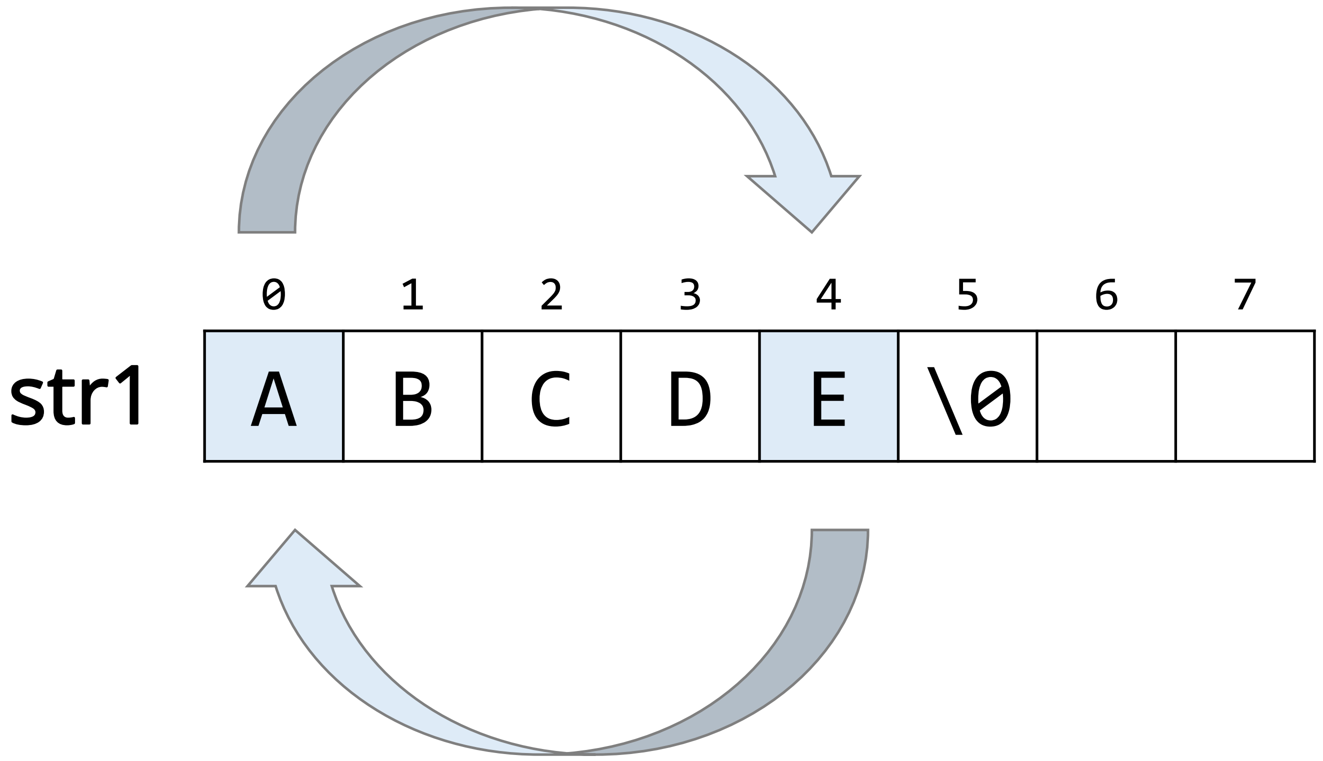
Task 4: Reversing a string

	0	1	2	3	4	5	6	7
str1	A	B	C	D	E	\0		

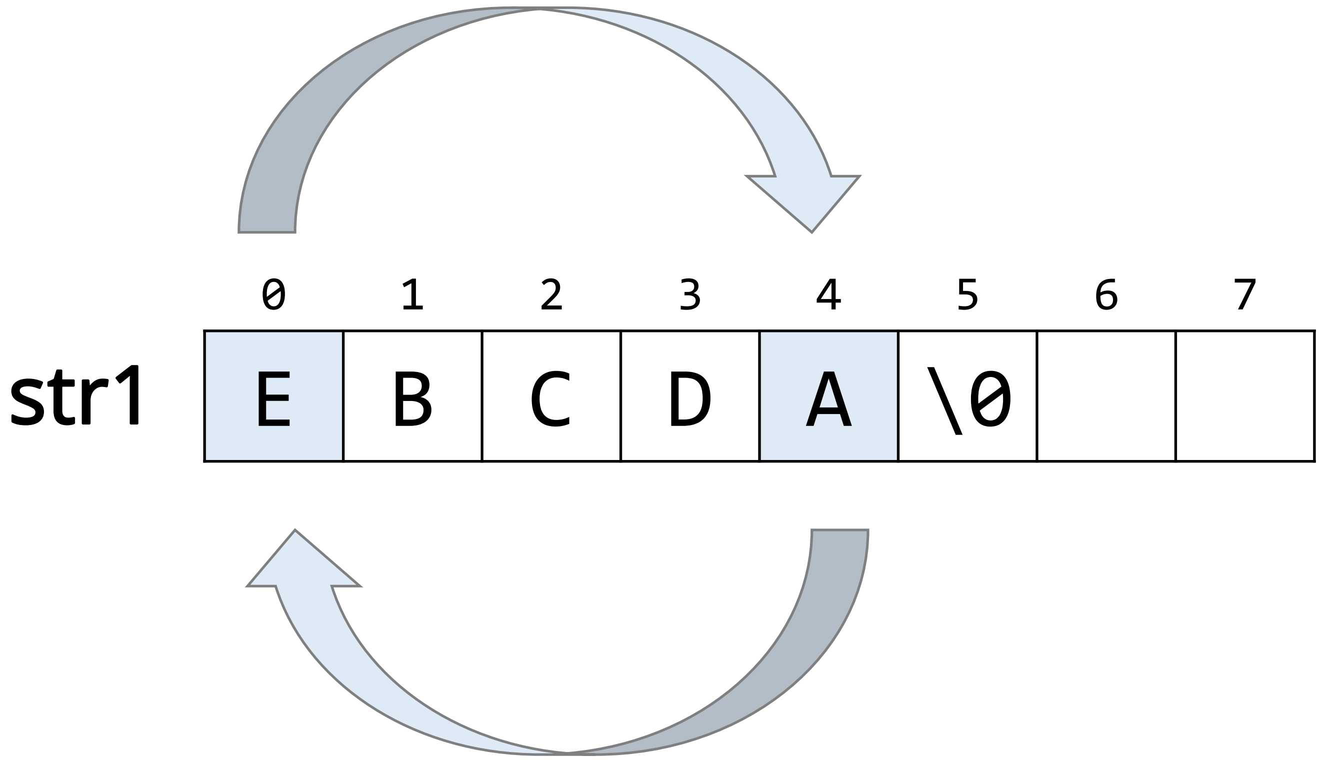
Task 4: Reversing a string



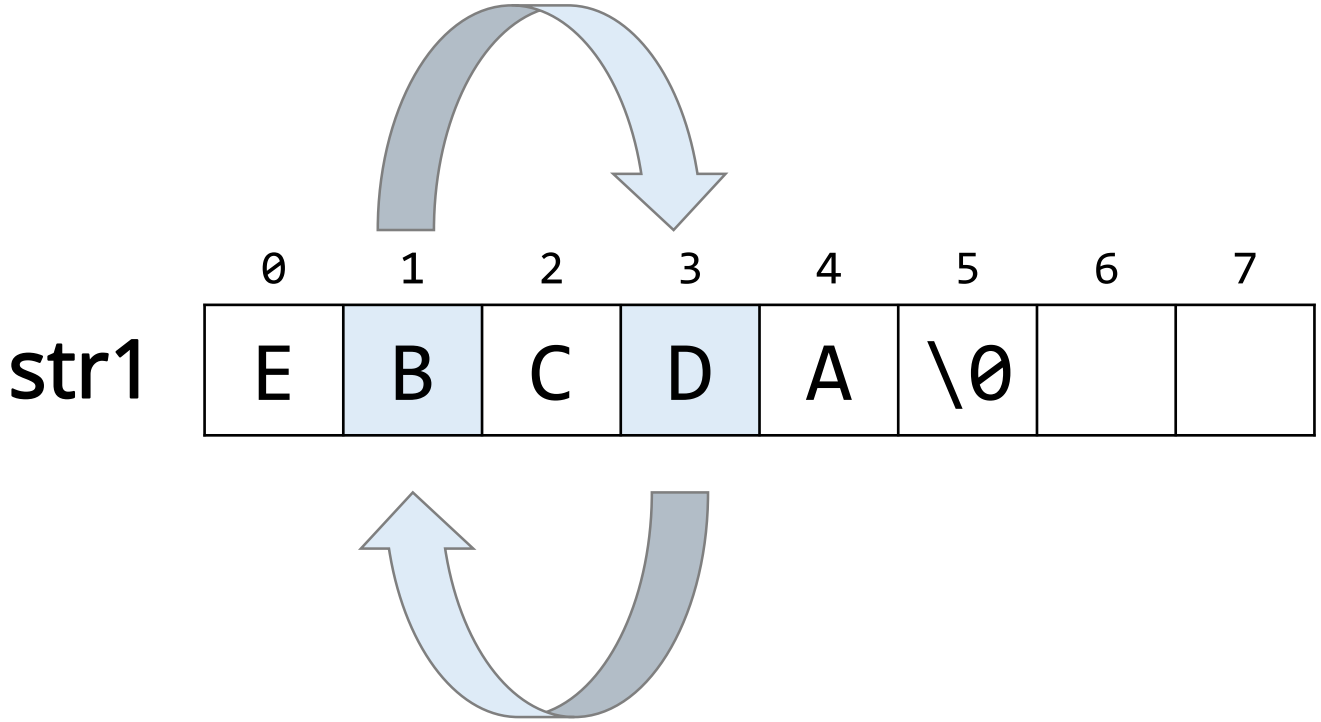
Task 4: Reversing a string



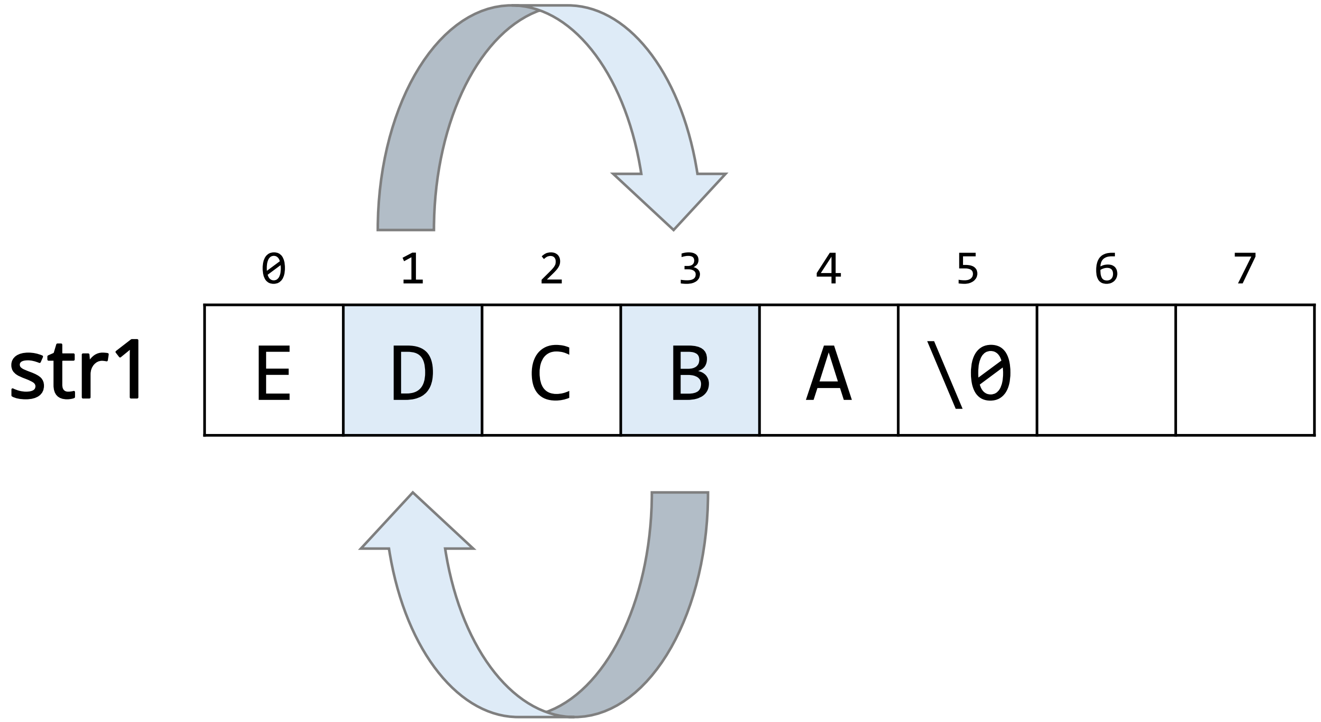
Task 4: Reversing a string



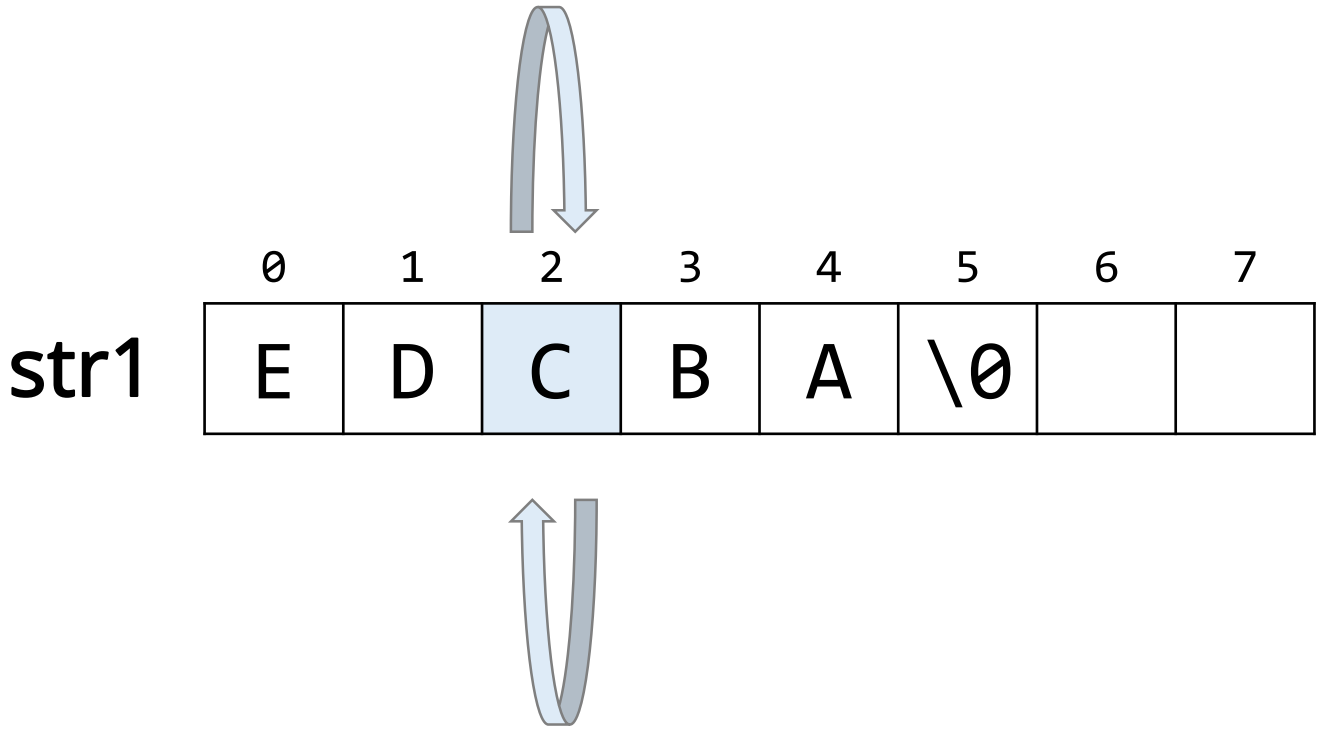
Task 4: Reversing a string



Task 4: Reversing a string



Task 4: Reversing a string



Task 4: Reversing a string

	0	1	2	3	4	5	6	7
str1	A	B	C	D	E	\0		

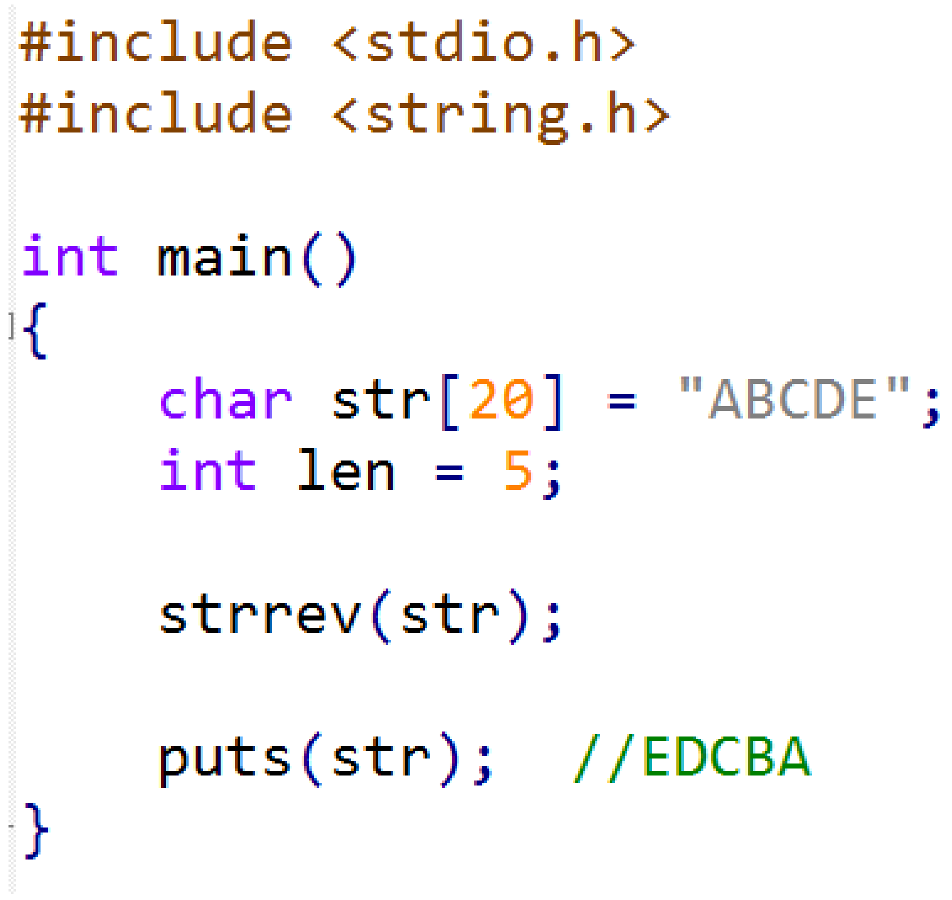
```
int len1 = strlen(str1);
```

- How many times should the loop run?

  len1 times?

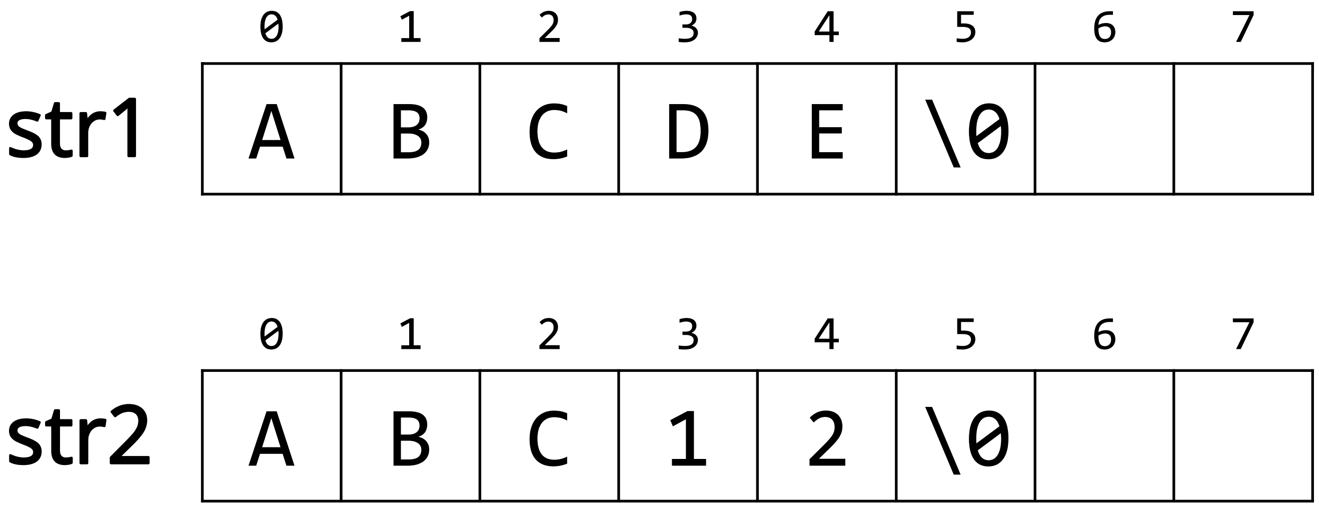
Task 4: Reversing a string

We can also use the library function

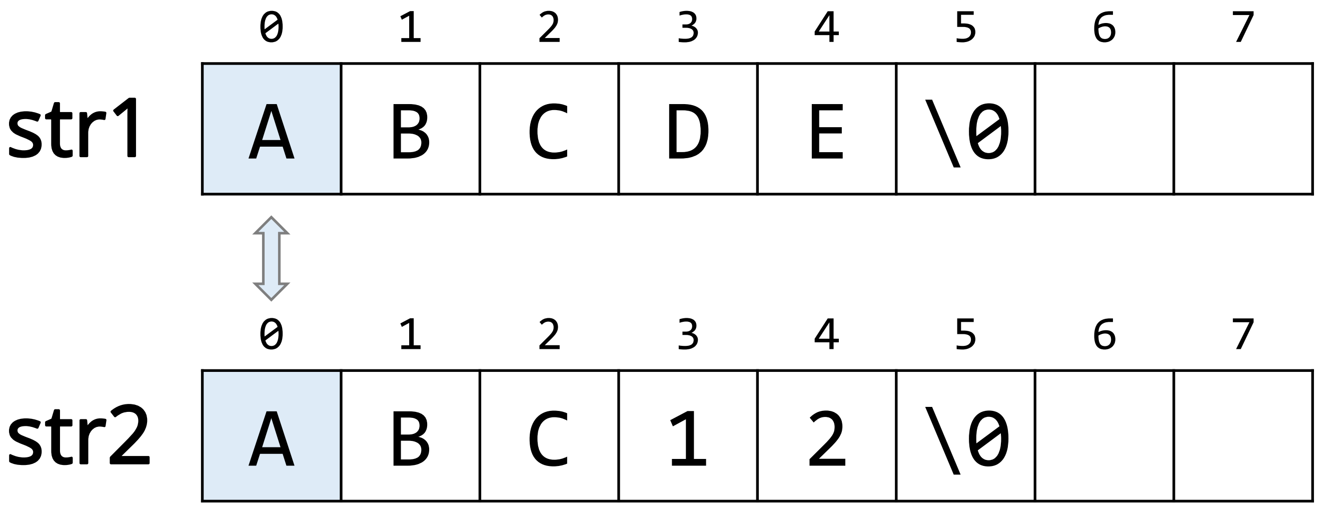


```c
#include <stdio.h>
#include <string.h>

int main()
{
    char str[20] = "ABCDE";
    int len = 5;

    strrev(str);

    puts(str);  //EDCBA
}
```

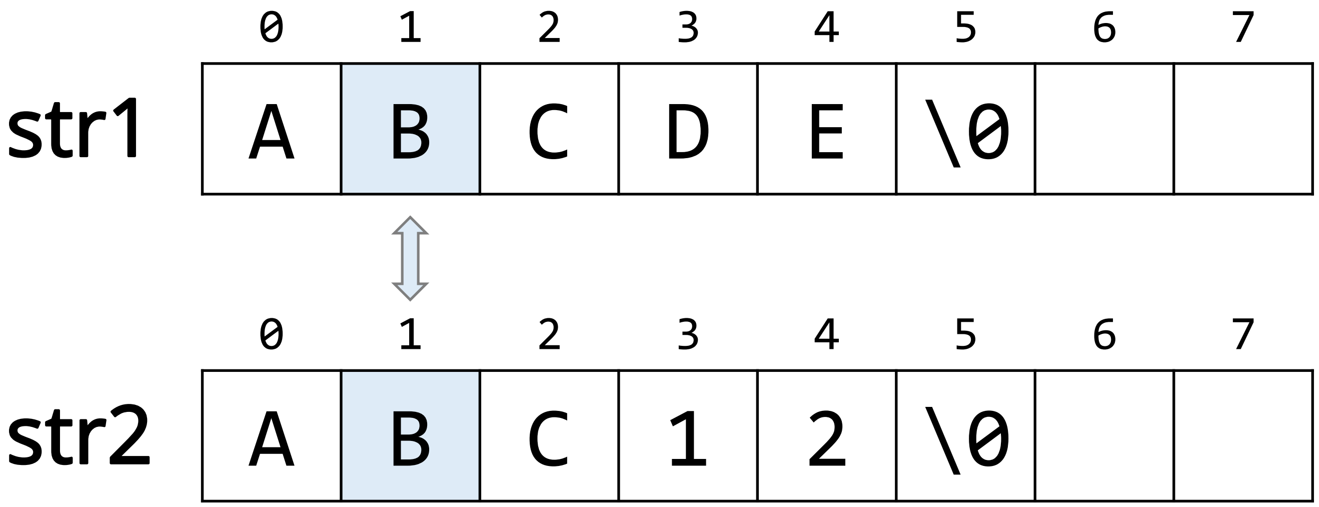
Task 5: Checking is two strings are equal



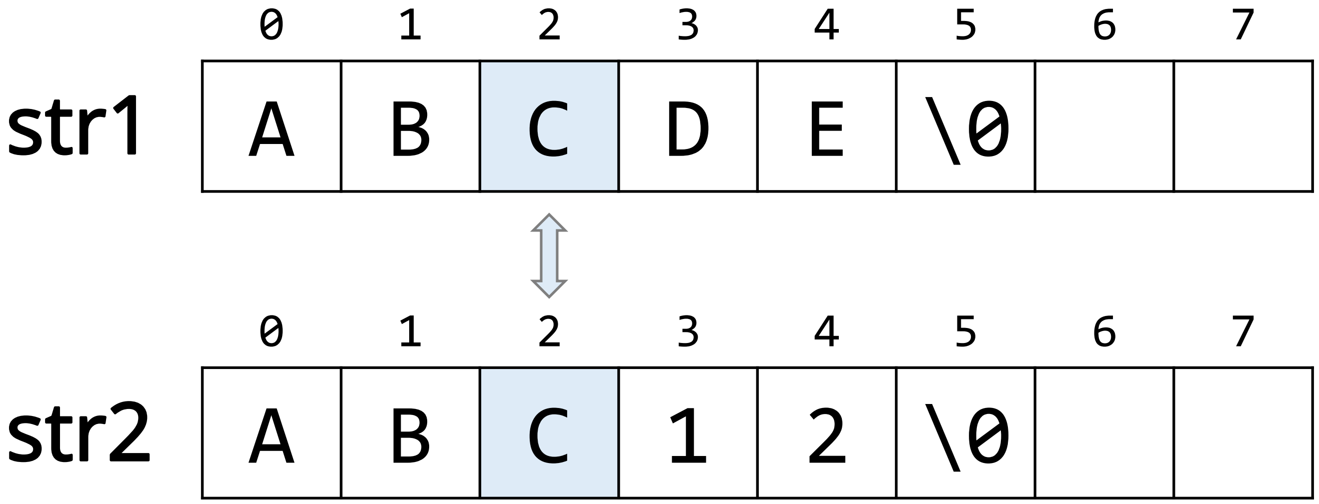
Task 5: Checking is two strings are equal



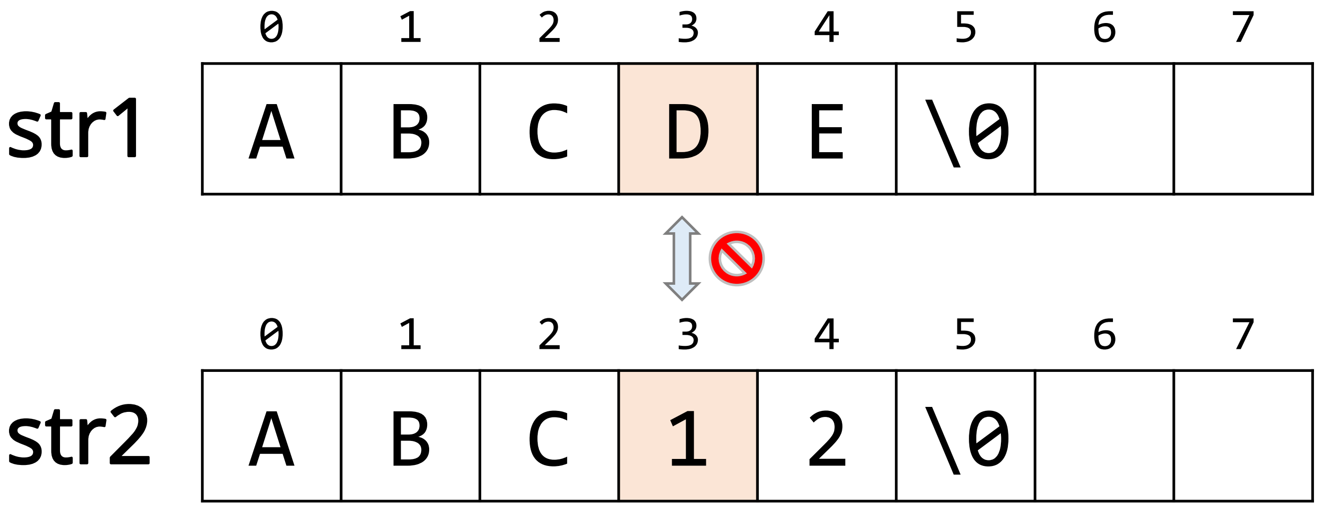
Task 5: Checking is two strings are equal



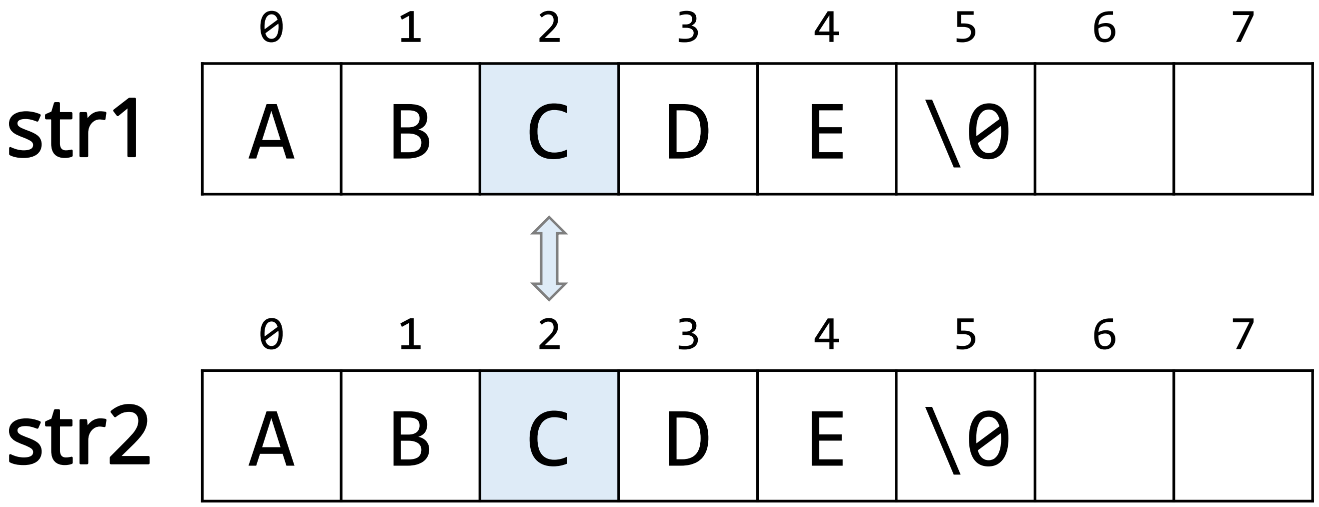
Task 5: Checking is two strings are equal



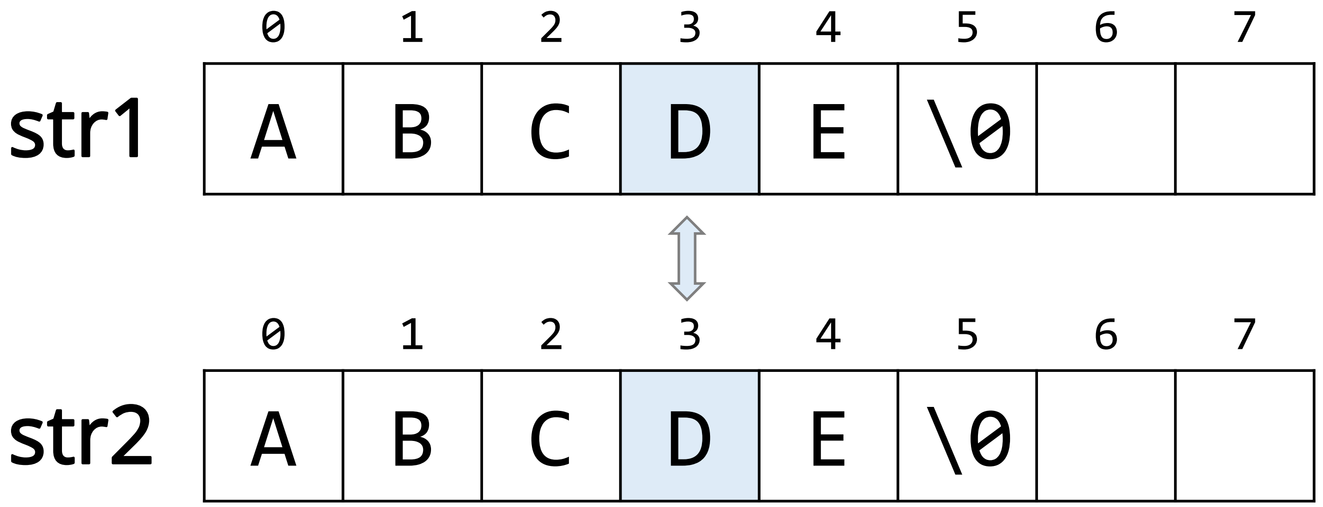
Task 5: Checking is two strings are equal



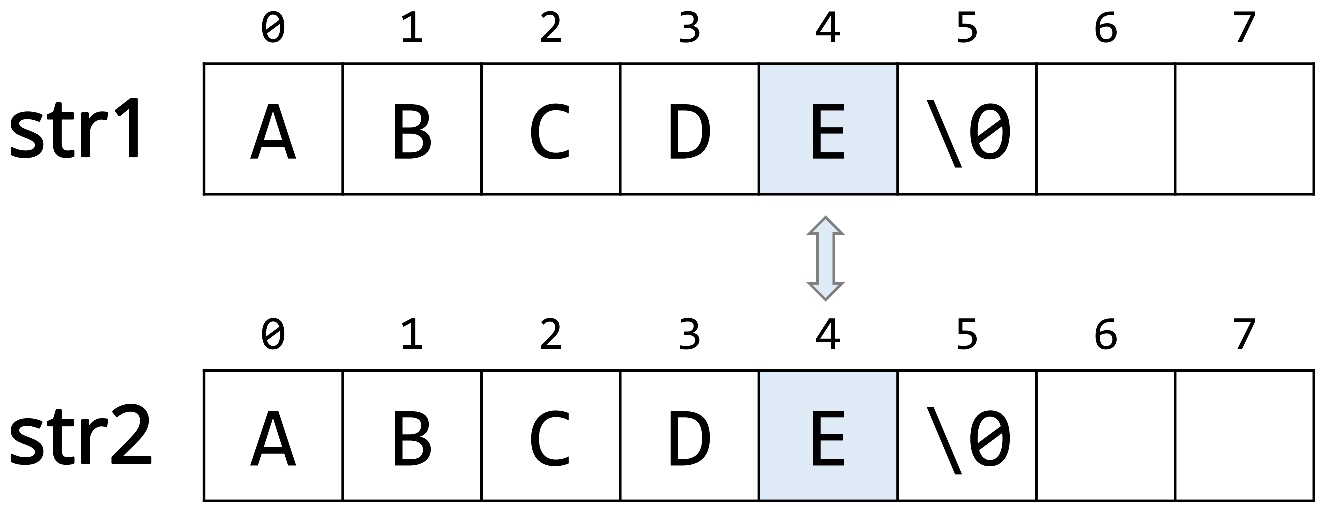
Task 5: Checking is two strings are equal



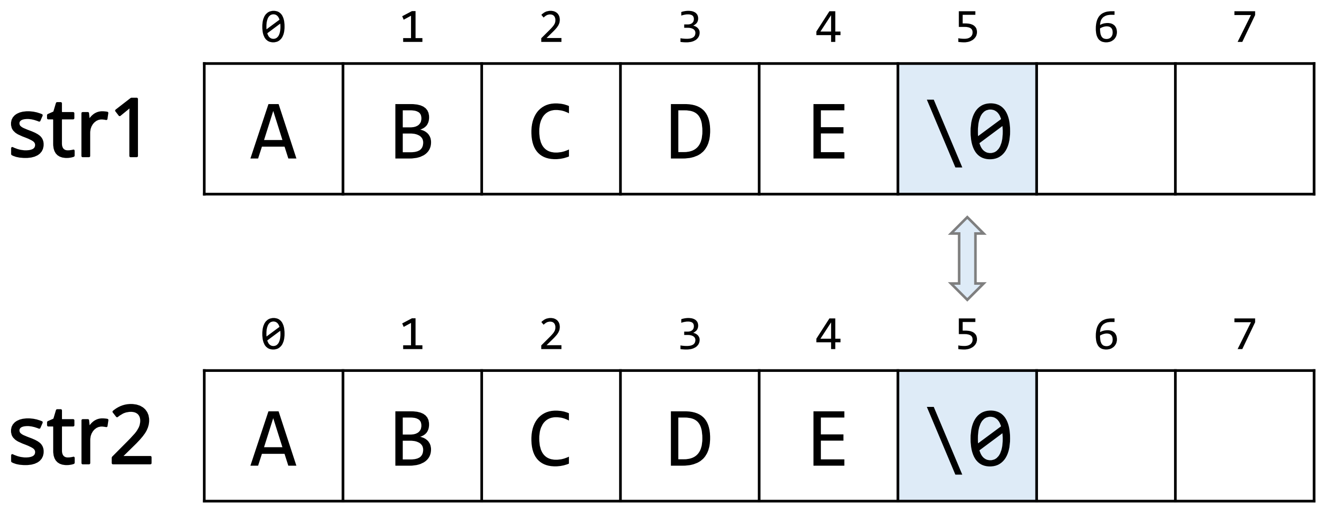
Task 5: Checking is two strings are equal



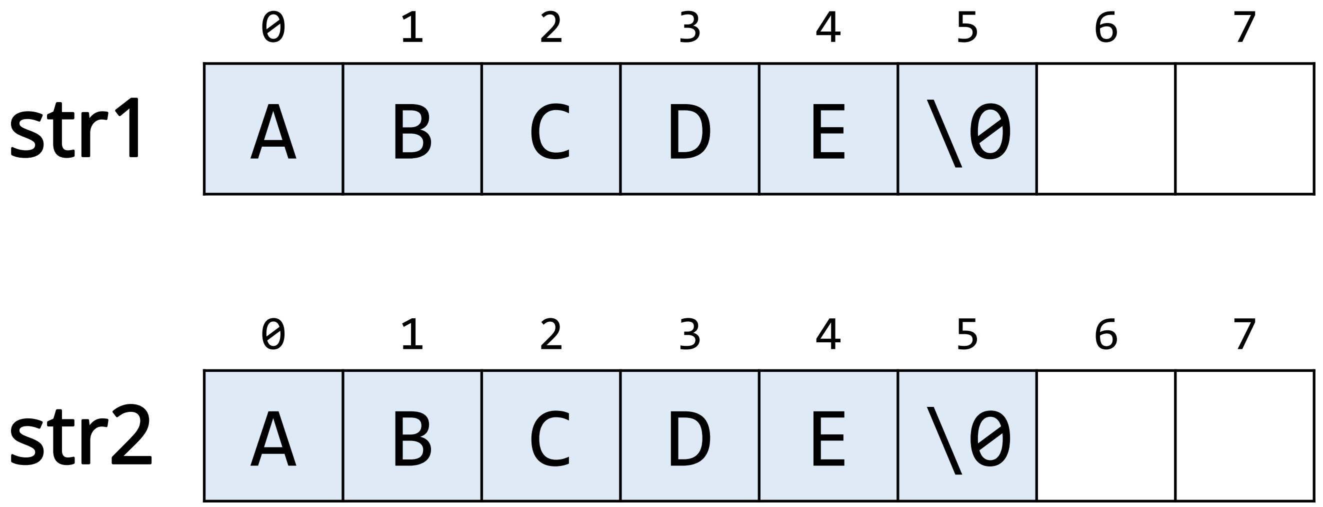
Task 5: Checking is two strings are equal



Task 5: Checking is two strings are equal

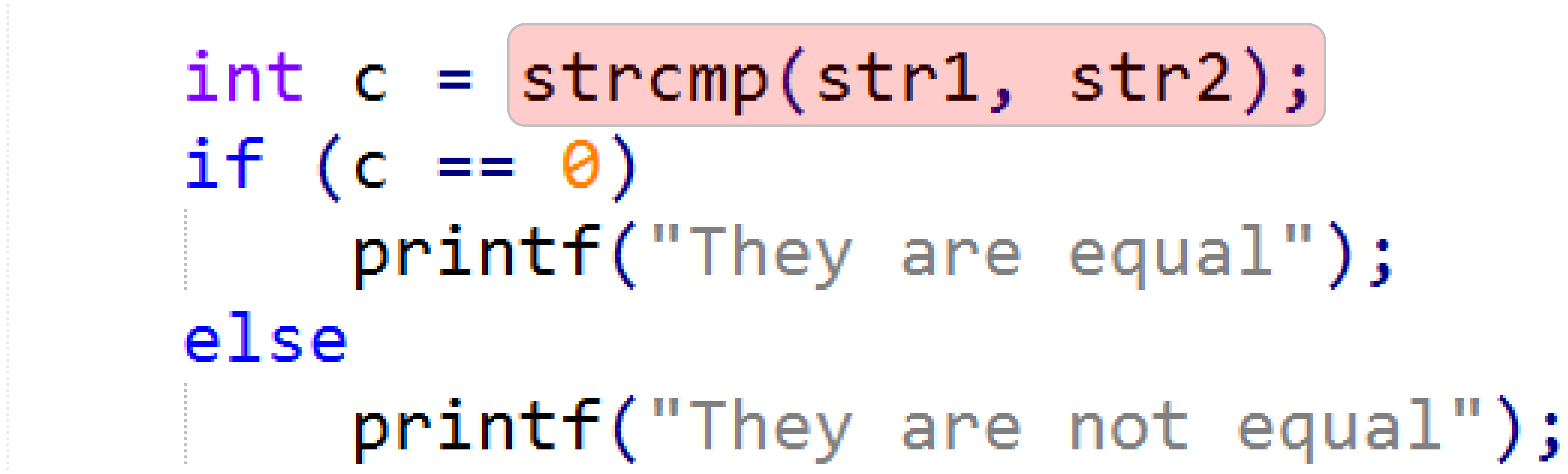


Task 5: Checking is two strings are equal



Task 5: Checking is two strings are equal

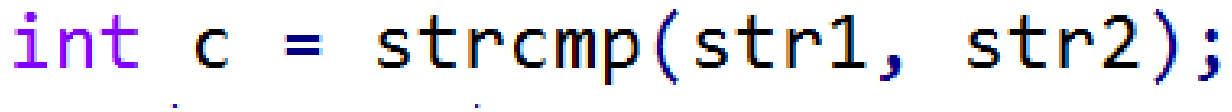
As usual, we can also use the library function



```c
int c = strcmp(str1, str2);
if (c == 0)
    printf("They are equal");
else
    printf("They are not equal");
```

Task 5: Checking is two strings are equal

As usual, we can also use the library function



```
int c = strcmp(str1, str2);
```

c < 0 means str1 is less than str2

Which means,

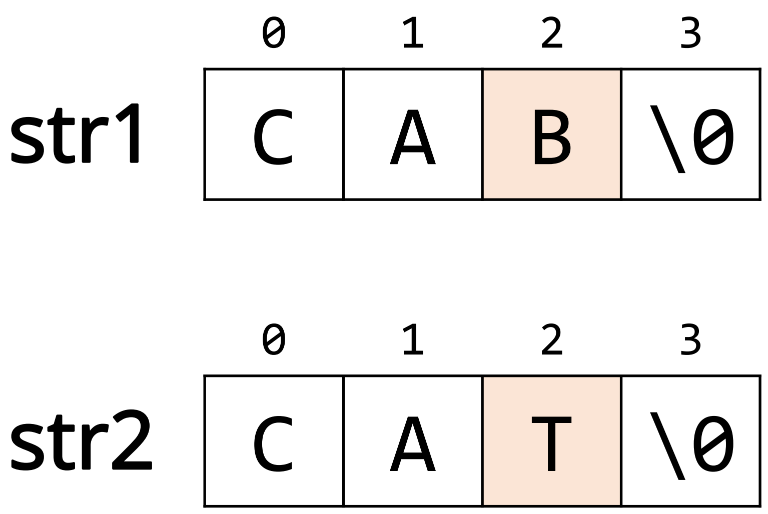
In dictionary, str1 comes before str2

Task 5: Checking is two strings are equal

As usual, we can also use the library function

```
int c = strcmp(str1, str2);
```

c < 0 means str1 is less than str2



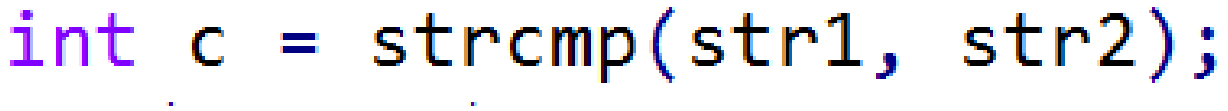
Which means,

In dictionary, str1 comes before str2

*str1 appears before str2 in **lexicographic order.**

Task 5: Checking is two strings are equal

As usual, we can also use the library function



```
int c = strcmp(str1, str2);
```

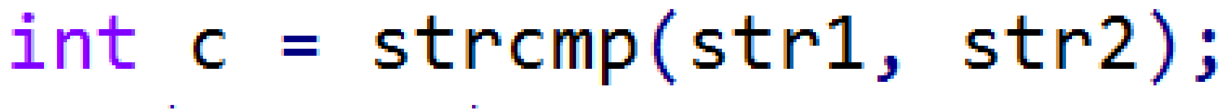
c > 0 means str1 is greater than str2

Which means,

In dictionary, str1 comes <u>after</u> str2

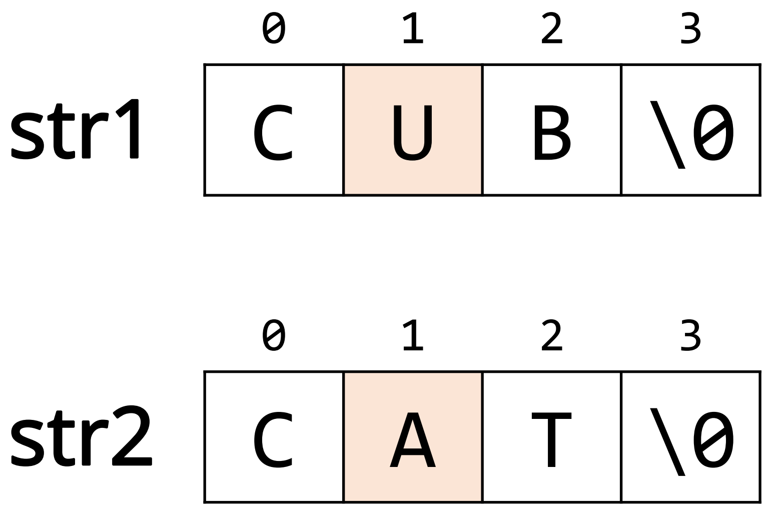
Task 5: Checking is two strings are equal

As usual, we can also use the library function



```c
int c = strcmp(str1, str2);
```

c > 0 means str1 is greater than str2



Which means,

In dictionary, str1 comes <u>after</u> str2