What is constant?

As the name suggests – something that never change

Once defined cannot be modified later in the code.

Defining constants

using #define

using const

Using const

Syntax: `const some_data_type some_variable_name`

Using #define

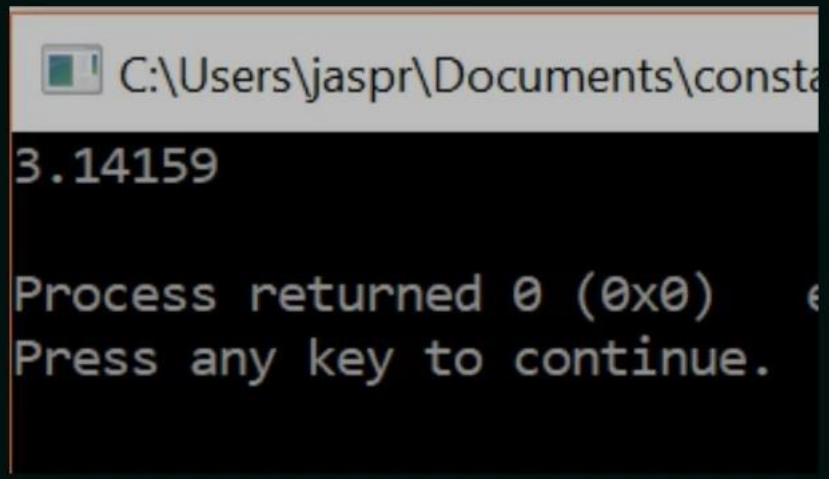
Also called Macro

Syntax: #define NAME value

Job of preprocessor (not compiler) to replace NAME with value.

```
#include <stdio.h>
#define PI 3.14159

int main() {
    printf("%.5f", PI);
    return 0;
}
```



Takeaways

1. Please don't add semicolon at the end.

```c
#include <stdio.h>
#define PI 3.14159;

int main() {
    printf("%.5f", PI);
    return 0;
}
```

Takeaways

2 Choosing capital letters for NAME is good practice.

```c
#include <stdio.h>
#define value 89

int main() {
    int value = 74;
    printf("%d", value);
    return 0;
}
```

≡

```c
#include <stdio.h>
#define value

int main() {
    int 89 = 74;
    printf("%d", value);
    return 0;
}
```

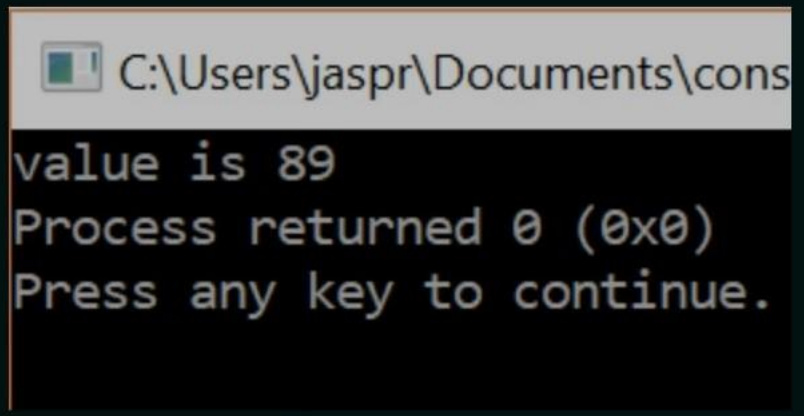
Error!

Takeaways

3 Whatever inside double quotes " " won't get replaced.

```c
#include <stdio.h>
#define value 89

int main() {
    printf("value is %d", value);
    return 0;
}
```



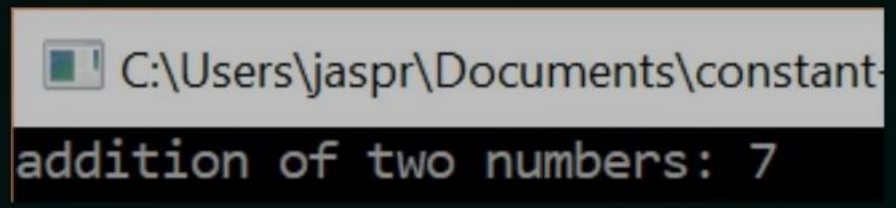
This is not replaced with 89

Takeaways

4 We can use macros like functions.

```c
#include <stdio.h>
#define add(x, y) x+y

int main() {
    printf("addition of two numbers: %d", add(4, 3));
    return 0;
}
```



Takeaways

5 We can write multiple lines using \

```
#include <stdio.h>
#define greater(x, y) if(x > y) \
                        printf("%d is greater than %d", x, y); \
                      else \
                        printf("%d is lesser than %d", x, y);
int main() {
    greater(5, 6);
    return 0;
}
```

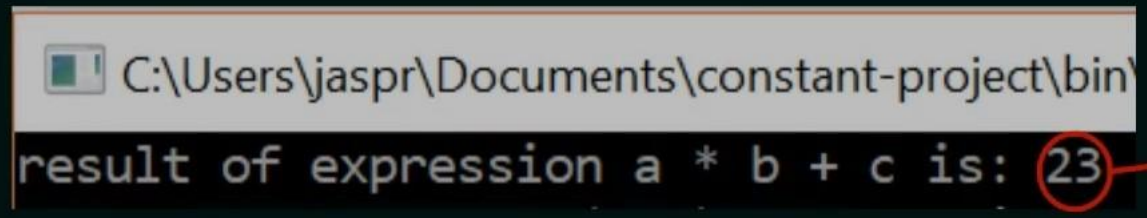
```
C:\Users\jaspr\Documents\constant-pr
5 is lesser than 6
```

Takeaways

6 First expansion then evaluation.

```c
#include <stdio.h>
#define add(x, y) x+y

int main() {
    printf("result of expression a * b + c is: %d", 5 * add(4, 3));
    return 0;
}
```



Because of 5 * 4 + 3.
Not 5 * 7

Takeaways

7 Some predefined macros like __DATE__ , __TIME__ can print current date and time.

```c
#include <stdio.h>

int main() {
    printf("Date: %s\n", __DATE__);
    printf("Time: %s\n", __TIME__);
    return 0;
}
```