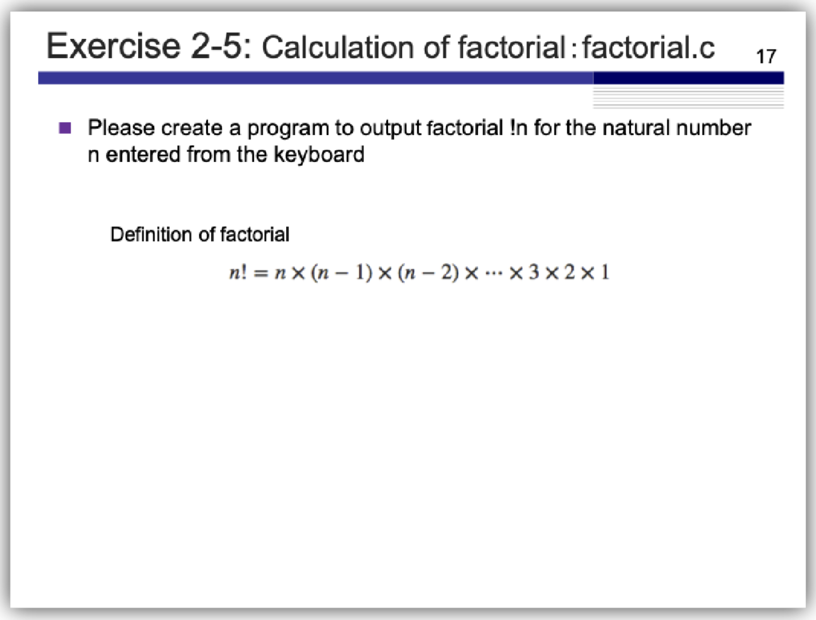
Exercise 2-5: factorial.c

Maximilian Fernaldy - C2TB1702



Exercise 2-5: Calculation of factorial : factorial.c

- Please create a program to output factorial !n for the natural number n entered from the keyboard

Definition of factorial

$$n! = n \times (n-1) \times (n-2) \times \cdots \times 3 \times 2 \times 1$$

Exercise 2-5 is another implementation of the for loop. Again there are multiple ways to go about this. A factorial can be defined as such:

$$n! = n \times (n-1) \times (n-2) \times \ldots \times 1$$

However, since we know that multiplication is commutative, we can also define it like this:

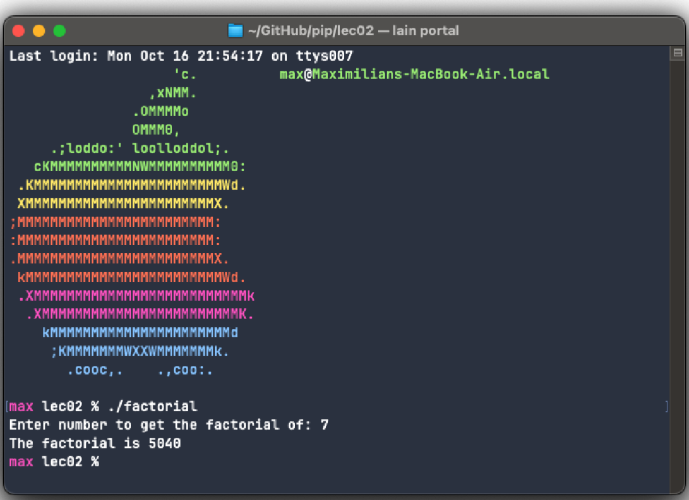
$$n! = 1 \times \ldots \times (n-2) \times (n-1) \times n$$

Which means our for loop can go both ways: going up from 1 and ending with n or going down starting with n and ending with 1. Ultimately both ways have their different use cases for specialized use, but since we only need the final result of the factorial for this exercise, either way is fine. I chose to go with the descending route.

```c
#include <stdio.h>

int main()
{
    /**** variable declaration ****/
    int n,factorial = 1; // Initialize factorial as 1 so that it defaults to 1 when 0 or 1 is entered

    printf("Enter number to get the factorial of: ");
    scanf("%d", &n); // get user input for the desired factorial

    /**** processing contents ****/
    for(;n>1;n--) {
        factorial *= n;
    }

    printf("The factorial is %d\n", factorial);

    return 0;
}
```

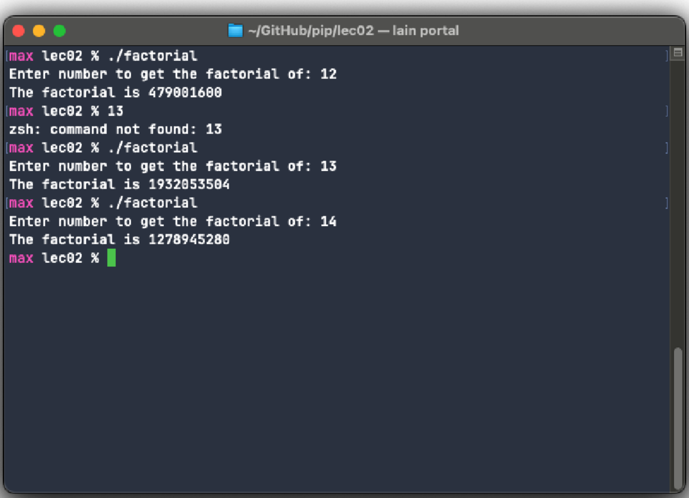
We omit the first field in the `for` loop because we have defined `n` using `scanf()`. Then we select `n>1` for the iteration condition, because we want to keep multiplying until `n` is equal to 1. Then we print the final result with `printf`.

Compiling and running the code yields the following:

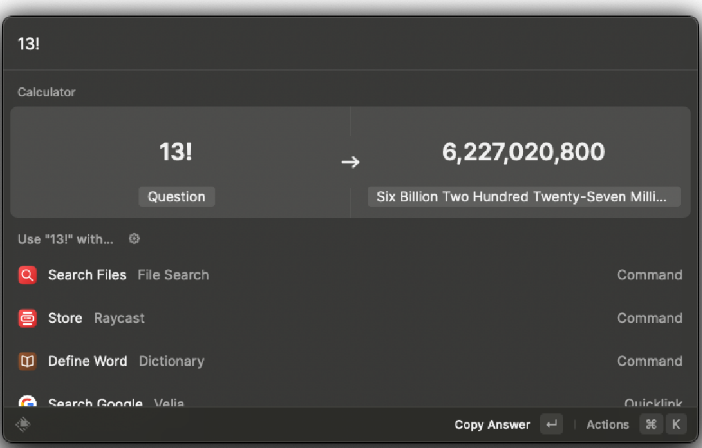


```
~/GitHub/pip/lec02 — lain portal
Last login: Mon Oct 16 21:54:17 on ttys007
                    'c.          max@Maximilians-MacBook-Air.local
                 ,xNMM.
               .OMMMMo
               OMMM0,
     .;loddo:' loolloddol;.
   cKMMMMMMMMMMNWMMMMMMMMMM0:
 .KMMMMMMMMMMMMMMMMMMMMMMMWd.
 XMMMMMMMMMMMMMMMMMMMMMMMX.
;MMMMMMMMMMMMMMMMMMMMMMMM:
:MMMMMMMMMMMMMMMMMMMMMMMM:
.MMMMMMMMMMMMMMMMMMMMMMMMX.
 kMMMMMMMMMMMMMMMMMMMMMMMMWd.
 .XMMMMMMMMMMMMMMMMMMMMMMMMMMk
  .XMMMMMMMMMMMMMMMMMMMMMMMMK.
    kMMMMMMMMMMMMMMMMMMMMMMd
     ;KMMMMMMMWXXWMMMMMMMk.
       .cooc,.    .,coo:.

max lec02 % ./factorial
Enter number to get the factorial of: 7
The factorial is 5040
max lec02 %
```

But factorials can get large very quick. Since an `int` type can only hold a value up to 2147483647, if we get values higher than that with a factorial, the output will be wrong. Trying to get higher and higher numbers, we see that at `n=13`, the program breaks. It outputs a value that is not equal to $12! \times 13$. A quick check with a calculator confirms this.



```
~/GitHub/pip/lec02 — lain portal
max lec02 % ./factorial
Enter number to get the factorial of: 12
The factorial is 479001600
max lec02 % 13
zsh: command not found: 13
max lec02 % ./factorial
Enter number to get the factorial of: 13
The factorial is 1932053504
max lec02 % ./factorial
Enter number to get the factorial of: 14
The factorial is 1278945280
max lec02 %
```



13!

Calculator

13! → 6,227,020,800

Question

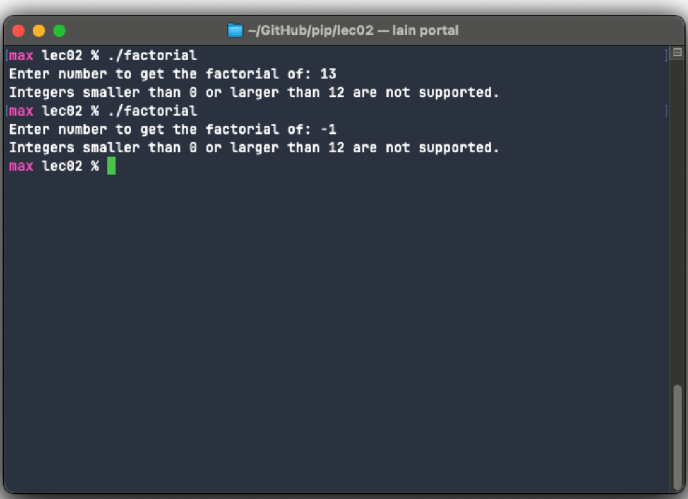
Six Billion Two Hundred Twenty-Seven Milli...

Use "13!" with...
Search Files File Search Command
Store Raycast Command
Define Word Dictionary Command
Search Google Velja Quicklink
Copy Answer
Actions

We don't want to give out false information, so to simply error handle this for now, we can just tell the user that factorials higher than 12 are not supported. Additionally we also add the condition for when `n` is smaller than 0.

```c
#include <stdio.h>

int main()
{
    /**** variable declaration ****/
    int n,factorial = 1; // Initialize factorial as 1 so that it defaults to 1 when 0 or 1 is entered

    printf("Enter number to get the factorial of: ");
    scanf("%d", &n); // get user input for the desired factorial
    if (n < 0 || n > 12) {
        printf("Integers smaller than 0 or larger than 12 are not supported.\n");
    }
    else {
        /**** processing contents ****/
        for(;n>1;n--) {
            factorial *= n;
        }

        printf("The factorial is %d\n", factorial);
    }
    return 0;
}
```

Now if we input numbers that don't satisfy the condition the program will alert the user that it is not supported.



```
max lec02 % ./factorial
Enter number to get the factorial of: 13
Integers smaller than 0 or larger than 12 are not supported.
max lec02 % ./factorial
Enter number to get the factorial of: -1
Integers smaller than 0 or larger than 12 are not supported.
max lec02 %
```

Another route we can go is to use another type, namely long integers, to do the factorials with. However since this is not in the scope of the assignment I will leave it be.