

TP3 Loop Statements

Ex 3.1 Display

What does the following program display? (Show Execution History)

```
int main ( ) {
int A=3, B=7, C=5;
int Mystery ;
while (A) {
    A=A --;
    C=C++;
    }
printf ("A=%d, B=%d, C=%d \n", A, B, C);
Mystery = (A >((B>C)? B : C))? A : ((B>C)? B : C);
printf (" Mystery =%d \n", Mystery );
return 0; }
```

Ex 3.2 Sum

Write a program that asks the user to enter a positive integer *n*, then calculates and displays the sum of the factorials of all numbers less than *n*: $S = \sum_{k=1}^{n} k!$

Example: For n=1: S=1!

For n=2: S=1!+2! For n=3: S=1!+2!+3!

Ex 3.3 Fibonacci sequence

Write a program that asks the user to enter a positive integer *n* then calculates and displays the *nth* term of the Fibonacci sequence.

The Fibonacci sequence is defined as follows:

$$\left\{ \begin{array}{ll} U_0 = 0, \\ U_1 = 1, \\ U_n = U_{n-1} + U_{n-2} \quad \text{for } n \geq 2 \end{array} \right.$$

For example, *U6* = 8 because *U6* = *U5* + *U4* = 5 + 3 = 8.

Ex 3.4 Square root

The square root of a positive real **A** can be obtained by an iterative method using the following recurring sequence

$$\begin{cases} U_0 = A/2, \\ U_n = \left(U_{n-1} + \frac{A}{U_{n-1}} \right) / 2 \end{cases}$$

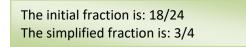
which converges to the square root of **A**. The calculation stops when the following condition is satisfied:

$$\left|\frac{U_{n+1}-U_n}{U_{n+1}}\right| \le \mathcal{E} \quad (\mathcal{E} \text{ is given})$$

Ex 3.5 Fraction Simplification

Write a program in C language that allows the user to enter a fraction in "numerator/denominator" form and displays the simplified fraction. You can use Euclid's algorithm to find the greatest common divisor (**GCD**) needed to simplify the fraction.

The program will display the initial fraction and the simplified fraction. For example:



Make sure you correctly handle cases where the user enters non-integer values, null fractions, or fractions with a zero denominator.

Ex 3.6 Multiplication Table

1. Write a program that displays the multiplication table for an arbitrary number n (given by the user). *Example:*

for n = 7: 1 * 7 = 7 2 * 7 = 14 . 9 * 7 = 63 10 * 7 = 70

2. Modify your program so that it displays a multiplication table in the following format:

X*Y	I	0	1	2	3	4	5	6	7	8	9	10	
0		0	0	0	0	0	0	0	0	0	0	0	
1		0	1	2	3	4	5	6	7	8	9	10	
2		0	2	4	6	8	10	12	14	16	18	20	
3		0	3	6	9	12	15	18	21	24	27	30	
4		0	4	8	12	16	20	24	28	32	36	40	
5		0	5	10	15	20	25	30	35	40	45	50	
6		0	6	12	18	24	30	36	42	48	54	60	
7		0	7	14	21	28	35	42	49	56	63	70	
8		0	8	16	24	32	40	48	56	64	72	80	
9		0	9	18	27	36	45	54	63	72	81	90	
10	T	0	10	20	30	40	50	60	70	80	90	100	

Ex 3.7 Numerical Patterns

Write a program that requests an integer N from the user and then displays the following numerical patterns: (Exp. N=4)

(A)	(B)	(C)
1234	1	1
123	2 3	25
12	4 5 6	368
1	7 8 9 10	4 7 9 10