

# C1- Unix & C Lab Seminar

pool\_c\_d03

Day 03

First Steps





## Instructions

### Before You Go Further

- Turn in directory pool\_c\_dO3.
- Respecting the norm takes time, but it is good for you. This way your code will respect the norm from the first written line to the last.
  - Read carefully the norm documentation. You should type "alt+i" instead of "tab"
- Do not care about the header. You don't have to add it into your file.
- You shall leave in your directory no other files than those explicitly specified by the exercices. If one of your files prevents the compilation with \*.c, the robot will not be able to do the correction and you will have a O.
- You might have to use the NULL keyword. This keywork is located in seddef.h
- Do not turn-in a main() function unless it has been explicitly asked.
- You are only allowed to use the write function to do the following exercices.



Remember it is always better to create your repository at the beginning of the day and to turn-in your work on a regular basis. Do not forget the permissions rights!





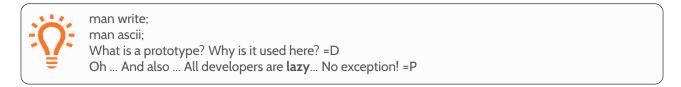
## Alpha (2pts)

**Turn in:** pool\_c\_dO3/ex\_O1/alpha.c **Prototype:** void alpha();

Write a function that displays the alphabet in uppercase on a single line, in ascending order from the letter 'A' followed by a '\n'

#### Example:

```
void alpha();
int main()
{
    alpha();
    return (0);
}
```







## Reverse Alpha (1pt)

Turn in: pool\_c\_dO3/ex\_O2/revalpha.c

#### Prototype: void revalpha();

Write a function that displays the alphabet in uppercase on a single line, in descending order from the letter 'Z' followed by a '\n'

#### Example:

```
void revalpha();
int main()
{
  revalpha();
  return (0);
}
```

```
Terminal - + X

~/pool_c_d03> cc *.c -o ex02

~/pool_c_d03> ./ex02

ZYXWVUTSRQPONMLKJIHGFEDCBA

~/pool_c_d03>
```





## True loop (2pts)

**Turn in:** pool\_c\_dO3/ex\_O3/my\_true\_loop.c **Prototype:** void my\_true\_loop(unsigned int n);

Write a function 'my\_true\_loop', taking a number as parameter. This function will print a number of '+' equal to this parameter followed by a '\n'

#### Example:

```
void my_true_loop(unsigned int n);
int main()
{
   my_true_loop(5);
   return (0);
}
```





## Conditions (2pts)

**Turn in:** pool\_c\_dO3/ex\_O4/conditions.c **Prototype:** void conditions(int n);

Write a function 'conditions' taking a number as parameter. This function will print '+' if the number is greater than 0, '-' if the number is lesser than 0 and '0' otherwise.

#### Example:

```
void conditions(int n);
int main()
{
  conditions(-564);
  conditions(564);
  conditions(0);
  return (0);
}
```





## my\_aff\_comb (3pts)

**Turn in:** pool\_c\_dO3/ex\_O5/my\_aff\_comb.c **Prototype:** int my\_aff\_comb();

Write a function that displays in the ascending order all the different combinations of three different digits in the ascending order.

#### Example:

#### main.c:

```
int     my_aff_comb();
int     main()
{
     my_aff_comb();
     return (0);
}
```

```
Terminal - + x

~/pool_c_d03> cc *.c -o ex05

~/pool_c_d03> ./ex05

012, 013, 014, 015, 016, 017, 018, 019, 023, ...., 789~/pool_c_d03>
```

In this example, all results are not shown but, of course, you have to display all different combinations. 987 is not here because 789 is already there.

999 is not here because that number's digits are not all different from each other.





### my\_putnbr (5pts)

**Turn in:** pool\_c\_dO3/ex\_O6/my\_putnbr.c **Prototype:** void my\_putnbr(int n);

Write a function 'my\_putnbr' taking a number as parameter. This function will print this number on the standard output.

#### Example:

#### main.c:

```
void     my_putnbr(int n);
int     main()
{
     my_putnbr(42);
     my_putnbr(-42);
     return (0);
}
```

## Exercise 7

### my\_aff\_comb2 (2pts)

**Turn in:** pool\_c\_dO3/ex\_O7/my\_aff\_comb2.c **Prototype:** int my\_aff\_comb2();

Write a function that displays all the different combinations of two numbers between 0 and 99, in ascending order.

#### Example:

```
int     my_aff_comb2();
int     main()
{
     my_aff_comb2();
     return (0);
}
```

```
Terminal - + x

~/pool_c_d03> cc *.c -o ex07

~/pool_c_d03> ./ex07

00 01, 00 02, 00 03, 00 04, 00 05, ..., 01 99, 02 03, ..., 98 99~/pool_c_d03>
```





## my\_aff\_combn (3pts)

**Turn in:** pool\_c\_d03/ex\_08/my\_aff\_combn.c **Prototype:** int my\_aff\_combn(int n);

Write a function that displays all the different combinations of n digits in the ascending order like in the example.

#### Example for n = 2:

