Rush 03

- rush(5,3) should display :

```
$>./a.out
ABBBC
B   B
ABBBC
$>
```

- rush(5, 1) should display:

```
$>./a.out
ABBBC
$>
```

- rush(1, 1) should display:

```
$>./a.out
A
$>
```

- rush(1, 5) should display:

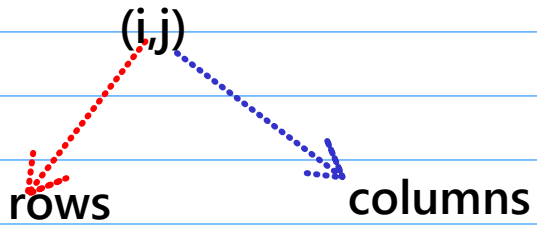
```
$>./a.out
A
B
B
B
A
$>
```

- rush(4, 4) should display:

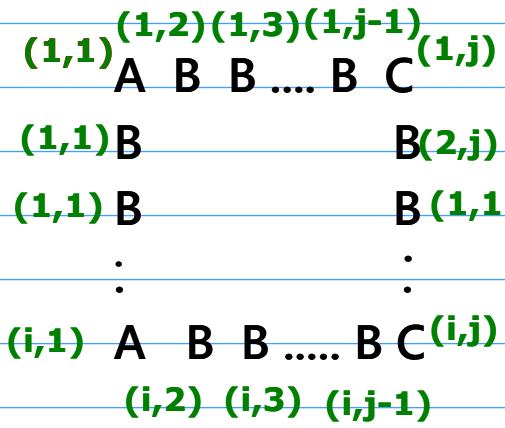
```
$>./a.out
ABBC
B  B
B  B
ABBC
$>
```

Solution

let's consider :



We can print patterns according to the number of rows and columns

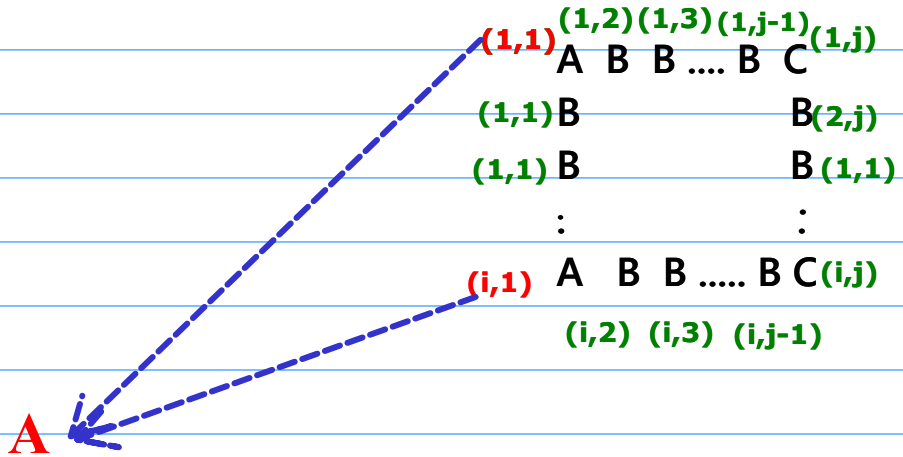


Then to print like this pattern,we have to use loops to control the indexes.

let's the number of rows to be print (i)=X

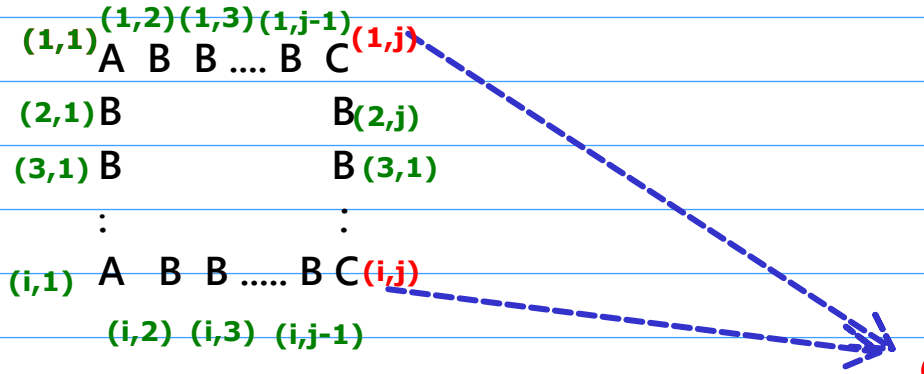
And columns (j)=Y

According from the given problem in ordert to print A



(1,1) or (x,1)

```
if ((i == 1 && j == 1) || (i == x && j == 1))
    ft_putchar('A');
```



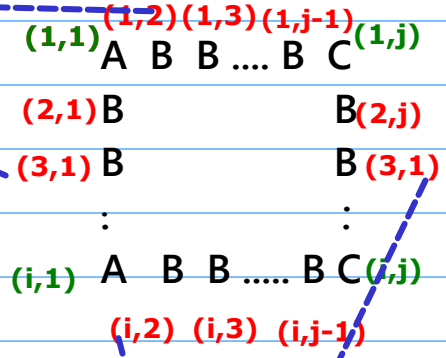
C

(1,y) or (x,y)

```
else if ((i == 1 && j == y) || (i == x && j == y))
    ft_putchar('C');
```

```
else if ((i == 1 && j != y && j != 1) || (j == 1 && i != x && i != 1))
    ft_putchar('B');
```

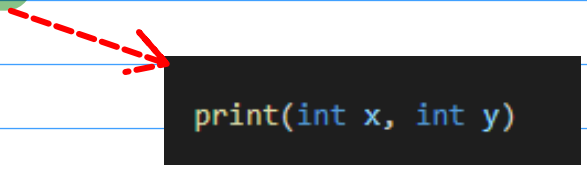
B



B

```
else if ((i == x && j != 1 && j != y) || (j == y && i != 1 && i != x))
    ft_putchar('B');
```

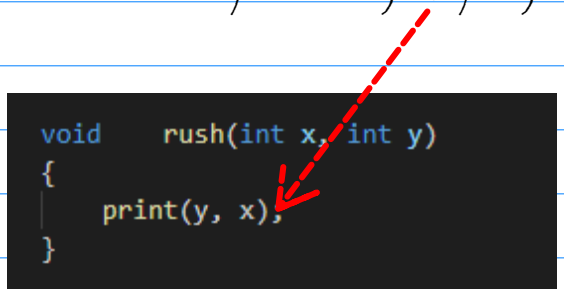
Then the output will be displayed under the function of print (x,y)



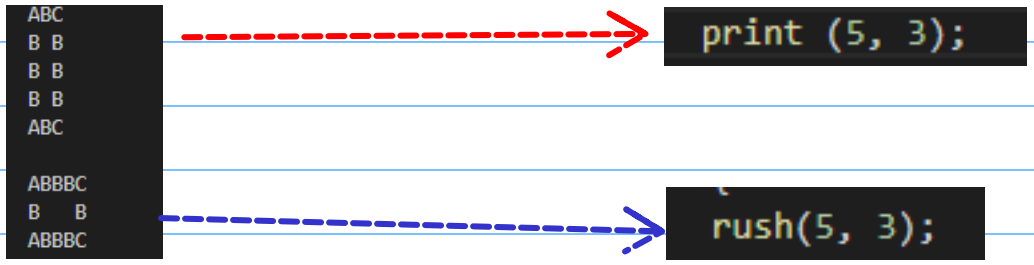
```
print(int x, int y)
```

- You must therefore create the function `rush` taking two variables of type `int` as arguments, named respectively `x` and `y`.
- Your function `rush` should display (on-screen) a rectangle of `x` characters for width, and `y` characters for length.

since the the rush function should be x for width and y for length, we have to reverse the components of x, y as follow



```
void    rush(int x, int y)
{
    print(y, x);
}
```

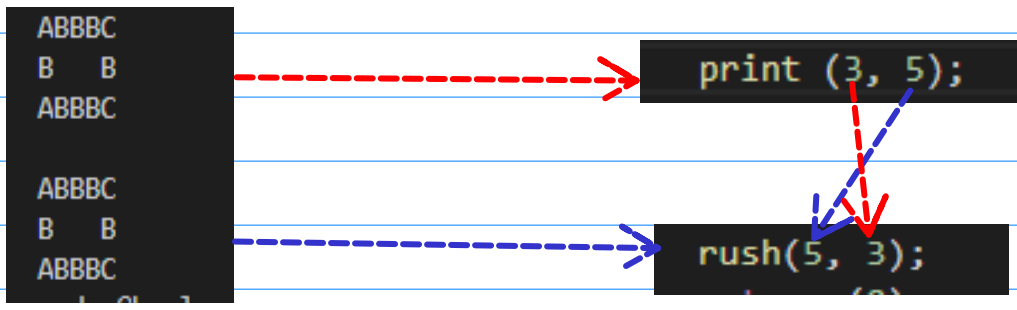


```
ABC
B B
B B
B B
ABC

ABBBC
B   B
ABBBC
```

```
print (5, 3);
```

```
rush(5, 3);
```



```
ABBBC
B   B
ABBBC

ABBBC
B   B
ABBBC
```

```
print (3, 5);
```

```
rush(5, 3);
```