

الـ Recursion بـ 3 خطوات

كيف تقرأوا أسئلة الركورسيا بدون ما تضيعوا

شرط التوقّف: الجواب معروف، بلا نداء `base case`

النداء: الدالة بتنادي حالها `recursive call`

1 شو يعني Recursion ؟

مثال: !n

دالة **بتنادي حالها**: `fact(5)` بتنادي `fact(4)` ، وهي بتنادي `fact(3)` ... لحد `fact(0)` .

شرط التوقّف: لما `n == 0` بترجّع `1` بلا نداء

النداء بيقرّب لشرط التوقّف: `n - 1`

النتيجة: `fact(5) = 5·4·3·2·1 = 120`

```
public static int fact(int n) {
    if (n == 0)
        return 1;
    return n * fact(n - 1);
}
```

2 3 خطوات لكل سؤال تتبّع

بتنفع بكل سؤال

1. **دوّروا على شرط التوقّف قبل كل إشي:** هين `n < 10` ، بترجّع `n` نفسه.
2. **كل نداء بسطر لحاله**، درجة تحت درجة. كل درجة بتستنّى جواب الدرجة اللي تحتها.
3. **وصلتوا لشرط التوقّف؟ وقّفوا بناء الدرج، وارجعوا لفوق** درجة درجة واحسبوا القيمة.

شو بترجّع `what(4072)` ؟

```
public static int what(int n) {
    if (n < 10)
        return n;
    return n % 10 + what(n / 10);
}
```

`n % 10` = آخر خانة، و `n / 10` = العدد بلا آخر خانة.

2 نازلين: بنبني الدرج

- `what(4072) = 2 + what(407)`
- `what(407) = 7 + what(40)`
- `what(40) = 0 + what(4)`
- `what(4)` = 4 — شرط التوقّف

3 طالعين: بنحسب القيمة

- `2 + 11 = 13`
- `7 + 4 = 11`
- `0 + 4 = 4`
- 4

الجواب: `what(4072) = 13` . يعني الدالة **بتجمع خانات العدد** (لكل `n >= 0`): `4 + 0 + 7 + 2 = 13`

الدرج هو الـ call stack: كل درجة نداء لسّا ما خلص، و**آخر نداء انفتح هو أوّل واحد بيخلص**.

بالصفحة 2: 5 أنماط بتتكرّر · بالصفحة 3: 4 أغلاط + سؤال تدريب ←

BagrutAce

5 أنماط بتتكرّر

1 قوّة

أرقام

احسبوا `b` بقوّة `e`.

```
public static int power(int b, int e) {
    if (e == 0)
        return 1;
    return b * power(b, e - 1);
}
```

`e` بينزل لـ `0`، وكل درجة **بتضرب بـ** `b` مرّة.

`power(2, 5) = 32`

2 قلب String

String

رجّعوا الـ String بالمقلوب.

```
public static String reverse(String s) {
    if (s.length() <= 1)
        return s;
    return reverse(s.substring(1))
           + s.charAt(0);
}
```

`s.substring(1)` = بلا أوّل حرف، و `s.charAt(0)` = أوّل حرف، **بنحطّه بالآخر**.

`reverse("java") = "avaj"`

3 مجموع array

array

اجمعوا كل الخانات، من `i` لآخر الـ array.

```
public static int sumArr(int[] a, int i) {
    if (i == a.length)
        return 0;
    return a[i] + sumArr(a, i + 1);
}
```

بنبلّش من `i = 0`، و `i == a.length` **يعني خلصت**.

`a = {4, 1, 7} → sumArr(a, 0) = 12`

4 عدّ مع if

بشرط

قدّيش مرّة الخانة `7` موجودة بالعدد؟

```
public static int count7(int n) {
    if (n == 0)
        return 0;
    if (n % 10 == 7)
        return 1 + count7(n / 10);
    return count7(n / 10);
}
```

بكل طريق في **return**: مع `1 +` لما الخانة `7`، وبدونه لما لأ.

`count7(7172) = 2`

5 طباعة: قبل النداء ولا بعده؟

فخ بأسئلة التتبّع

قبل النداء `down(3) → 3 2 1`

```
public static void down(int n) {
    if (n == 0)
        return;
    System.out.print(n + " ");
    down(n - 1);
}
```

بعد النداء `up(3) → 1 2 3`

```
public static void up(int n) {
    if (n == 0)
        return;
    up(n - 1);
    System.out.print(n + " ");
}
```

قبل النداء `down(3) print 3` → `down(2) print 2` → `down(1) print 1` → `down(0) stop`

بعد النداء `up(3)` → `up(2)` → `up(1)` → `up(0) stop` ↩ `print 1` → `print 2` → `print 3`

اللي **قبل** النداء بيصير واحنا نازلين عالدرج، واللي **بعد** النداء بيستنّى لحد ما نطلع. بسؤال «شو بيطبع؟» اسألوا أوّل إشي: **الطباعة قبل النداء ولا بعده؟**

بالصفحة 3: 4 أغلاط بتتكرّر + سؤال تدريب مع الحل ←

BagrutAce

4 أغلاط بتتكرّر

1 بلا شرط توقّف

```java
public static int fact(int n) {
    // no stop condition!
    return n * fact(n - 1);
}
```

`fact(3) → StackOverflowError`

النداءات ما بتخلص لحد ما الـ stack يتعبّى. **أوّل سطر بالدالة: شرط التوقّف.**

2 `n--` بدل `n - 1`

```java
public static int fact(int n) {
    if (n == 0)
        return 1;
    return n * fact(n--);
}
```

`fact(3) → StackOverflowError`

`n--` بيبعث `n` نفسه وبعدين بينقّص، فالنداء ما بيقرّب لـ `0`. اكتبوا `n - 1`.

3 قيمة النداء ضاعت

```java
public static int count7(int n) {
    if (n == 0)
        return 0;
    int c = 0;
    if (n % 10 == 7)
        c = 1;
    count7(n / 10);   // result lost
    return c;
}
```

`count7(777) → 1` مش `3`

الصح: `return c + count7(n / 10);`

وبلا `return c;` بالمرّة؟ ما بيعمل compile:
missing return statement

4 شرط توقّف متأخّر

```java
public static int sumArr(int[] a, int i) {
    if (i > a.length)
        return 0;
    return a[i] + sumArr(a, i + 1);
}
```

`sumArr(a, 0)` **بيوقّف بـ:**

`ArrayIndexOutOfBoundsException`

لما `i == a.length` ما في `a[i]`. لازم توقفوا **هناك بالزبط**: `if (i == a.length)`.

? سؤال تدريب زي البچروت

جرّبوا قبل ما تشوفوا الحل

```java
public static int mystery(int a, int b) {
    if (b == 0)
        return 0;
    return a + mystery(a, b - 1);
}
```

أ. شو بترجّع `mystery(4, 3)` ؟

ب. شو بتحسب الدالة لكل `a`، و`b` من `0` وفوق؟

حلّوا بالـ 3 خطوات: شرط التوقّف `b == 0`، ودرجة لكل نداء.

2 نازلين

`mystery(4, 3) = 4 + mystery(4, 2)`

`mystery(4, 2) = 4 + mystery(4, 1)`

`mystery(4, 1) = 4 + mystery(4, 0)`

`mystery(4, 0) = 0` — شرط التوقّف

3 طالعين

`4 + 0 = 4`

`4 + 4 = 8`

`4 + 8 = 12`

أ. `mystery(4, 3) = 12` **ب.** `a · b` يعني `a + a + … + a` (`b` مرّات)، يعني ضرب.

بدّكم تتدرّبوا على أسئلة Recursion؟

حلّوا على BagrutAce، وAce بيصلّحه إلكم على طول.

bagrutace.com ← يلّا نبلّش