Task 2: Environment Variables & Process Inheritance

شرح وتوثيق تفصيلي ثنائي اللغة (عربي / English) للمهمة الثانية - GitHub Documentation

> 🎯 **Objective | الهدف:** Prove whether a child process created via `fork()` inherits the environment variables from its parent process.

1. المفاهيم الأساسية | Core Concepts

1.1 What is `fork()`?

هو استدعاء نظام في أنظمة Linux/Unix ينشئ عملية جديدة بالكامل بأسلوب الاستنساخ. أمر `fork()` العملية المُنْشِئَة تُسمى **Parent Process** والعملية الجديدة المُنْشَأَة تُسمى **Child Process**.

1.2 Environment Variables & `environ` Array

متغيرات البيئة هي قيم حيوية يُعتمد عليها النظام. يتم تخزينها في الذاكرة ويمكن الوصول إليها عبر مصفوفة المؤشرات العامة `extern char **environ`.

2. الكود البرمجي | C Source Code Analysis

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

extern char **environ;

void printenv() {
    int i = 0;
    while (environ[i] != NULL) {
        printf("%s\n", environ[i]);
        i++;
    }
}

void main() {
    pid_t childPid;
    switch(childPid = fork()) {
        case 0:  /* child process */
            printenv();
            exit(0);
        default: /* parent process */
            // printenv();
            exit(0);
    }
}
```

3. خطوات التنفيذ والمقارنة | Implementation Steps

Step 1: طباعة مخرجات الابن في `file1`

```
gcc myprintenv.c -o myprintenv
./myprintenv > file1
cat file1
```

Step 2: التعديل بواسطة `nano` وطباعة مخرجات الأب في `file2`

```
nano myprintenv.c
# طفي printenv() تحت case 0 وفعلها تحت default
gcc myprintenv.c -o myprintenv
./myprintenv > file2
```

Step 3: المقارنة بواسطة `diff`

```
diff file1 file2
```

4. النتيجة والاستنتاج النهائي | Final Conclusion

> **النتيجة العملية:** عند تشغيل `diff file1 file2` تظهر الشاشة فارغة بدون أي مخرجات.
> **السبب العلمي:** الشاشة الفارغة تعني أن الملفين متطابقان بنسبة 100%. هذا يثبت أن الابن أخذ نسخة طبق الأصل من مصفوفة متغيرات البيئة الخاصة بالأب لحظة التنفيذ دون أي تعديل.