🧠 المنطق البرمجي: كيف "يفكر" البرنامج؟

في هذه المحاضرة، نضع القواعد التي تحكم تدفق البيانات، فإذا كانت البيانات هي "المادة الخام"، فإن المنطق البرمجي هو "نظام الإدارة" الذي يوجهها. الهدف هنا هو تمكين البرنامج من التفكير عبر **الشروط**، والعمل بجد عبر **الحلقات التكرارية** لمعالجة مئات التقارير في لمح البصر.

🚦 الجمل الشرطية (Conditions): مفرق الطرق النفسي

تخيل الجملة الشرطية كأنها مفرق طرق في التشخيص النفسي؛ حيث نستخدمها لتصنيف الحالات بناءً على معطيات محددة. في بايثون، نعتمد على ثلاث كلمات مفتاحية:

- **if (إذا):** لفحص الشرط الأول؛ إذا تحقق، ينفذ الحاسوب الأمر الملحق به.
- **elif (وإلا إذا):** لفحص شروط بديلة في حال عدم تحقق الشرط الأول.
- **else (وإلا):** وهي الملاذ الأخير الذي ينفذ إذا لم تتحقق أي من الشروط السابقة.

مثال تربوي (تصنيف القلق): إذا حصل طالب على درجة 8 في مقياس القلق، يبرمج الحاسوب ليقول: "قلق مرتفع يحتاج لتدخل فوري"، أما إذا كانت الدرجة 5 فيصنفها كـ "قلق متوسط"، وما دون ذلك فهو "حالة مستقرة".

⚖️ أدوات المقارنة والربط المنطقي

لكي يتمكن الحاسوب من اتخاذ القرار، يحتاج لرموز دقيقة للمقارنة بين البيانات:

- **المقارنة:** نستخدم == للتساوي تماماً، و != لعدم التساوي، و > أو < للمقارنة الكمية.
- **الربط:** نستخدم and لربط شرطين يجب تحققهما معاً، أو or إذا كان يكفي تحقق أحدهما فقط.

🔁 الحلقات التكرارية (Loops): المساعد الذي لا يمل

في البحث التربوي، قد نحتاج لفحص بيانات 1000 طالب؛ وبدلاً من كتابة كود لكل طالب، نستخدم الحلقات التكرارية التي تكرر الخطوات آلياً حتى انتهاء القائمة:

1. **حلقة for:** تستخدم للمرور على عناصر قائمة محددة (مثل قائمة أسماء الطلاب).
2. **حلقة while:** تستخدم لتكرار أمر ما طالما أن هناك شرطاً محققاً (مثل استمرار عرض الأسئلة حتى ينهي الطالب الاختبار).
3. **أداة break:** تستخدم لإيقاف التكرار فوراً عند العثور على مؤشر خطير (مثل رصد كلمة تدل على خطر الانتحار في نص).

🤝 القوة الحقيقية: دمج الشروط مع الحلقات

تتجلى عبقرية تحليل المحتوى (Content Analysis) عندما نضع "الشرط" داخل "الحلقة".

تطبيق عملي: يمكننا جعل الحاسوب يمر على مذكرات يومية لطالب، وإذا رصد كلمات مثل (ضربني، سخروا، تنمر)، يقوم بإرسال تحذير فوري للأخصائي بوجود "مؤشر قلق".

🚀 **لماذا هذا المسار حيوي للأخصائي النفسي؟**

البرمجة ليست مجرد كتابة أكواد، بل هي بناء منطق يساعد في:

- **تصحيح الاختبارات:** كود بسيط من 5 أسطر يمكنه تصحيح آلاف الاختبارات بدقة متناهية.
- **الملاحظة المنظمة:** تحليل الأنماط الزمنية للسلوك (مثل: هل يزداد العدوان بعد وقت الفسحة؟).
- **تخصيص التعلم:** بناء أنظمة ذكية توجه الطالب لمحتوى إثرائي إذا أخطأ في سؤال محدد.

الخلاصة: الشروط هي التي تعطي البرمجيات "ذكاءها" في الاختيار، والحلقات هي التي تعطيها "كفاءتها" في العمل. في المحاضرة القادمة، سنطبق هذا المنطق على ملفات نصية كاملة من جهاز الكمبيوتر لنعالجها آلياً.

💡 **مهمة للتفكير:** لو أردت تصميم برنامج يكتشف "التنمر الإلكتروني" في محادثات الطلاب، ما هي الشروط (if) التي ستضعها؟ وما هي الكلمات التي ستحتاج للمرور عليها باستخدام (for)؟