

جامعة حماة

كلية العلوم في مصيف / السنة الأولى

المادة: لغات البرمجة

المحاضرة الأولى: الخوارزميات Algorithms

العام الدراسي ٢٠١٩ - ٢٠٢٠





الحاسب الآلي Computer

- الحاسب الآلي : هو جهاز إلكتروني ينفذ الأوامر و التعليمات المكتوبة بإحدى لغات البرمجة و بسرعة عالية جداً تصل إلى عدة مليارات عملية في الثانية، و يعتمد مبدأ عمله على تلقي البيانات Data بمختلف أنواعها من وحدات الدخل Input Equipment's الموصولة به، فيقوم بمعالجتها Processing و إظهار النتائج (معلومات Information) على إحدى وحدات الإخراج Output Equipment's.
- يمكن الاستفادة من الحاسب في كافة مجالات الحياة (العلمية و الاقتصادية و الاجتماعية و الترفيهية)

Data → Processing → Information



Input



Output



مفاهيم أساسية Main Concepts

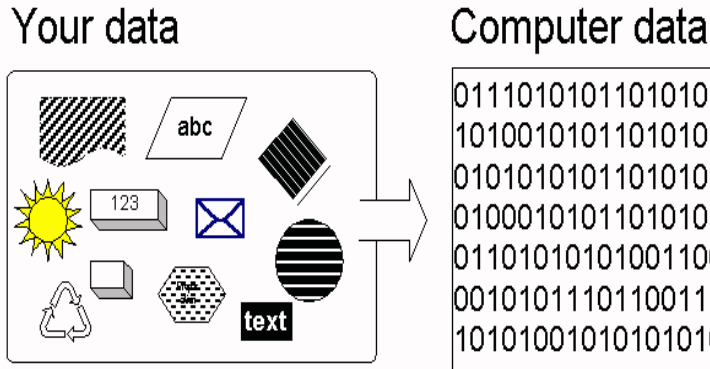
- **البيانات Data:** عبارة عن الكلمات (الحروف) والأرقام والرموز المتعلقة بموضوع معين مثل (أسماء الطلاب، أعمارهم، درجات المواد... الخ)، وهذه البيانات هي التي يتم إدخالها للحاسب ويتم عليها إجراء العمليات (معالجة البيانات) للحصول على المعلومات المفيدة.
- **المعلومات Information:** هي ناتج معالجة البيانات وبمعنى آخر فإن المعلومات هي الحقائق المنظمة والمفيدة الناتجة من معالجة البيانات فمثلا من البيانات السابقة عن الطلبة يمكن الحصول على المعلومات الآتية (مجموع الدرجات لكل طالب، ترتيب الطلبة حسب مجموع الدرجات).



البيانات و المعلومات

• مثال يوضح الفرق بين البيانات والمعلومات ومفهوم معالجة البيانات :

- تتسلم لجنة تصحيح الامتحانات في الجامعة البيانات الخاصة بالطلاب (أسماء الطلاب وأرقامهم ودرجاتهم في المواد المختلفة) ثم تقوم اللجنة بعمليات معالجة مختلفة مثل جمع درجات العملي و النظري للحصول على نتائج كل طالب على حدة ثم فرز الدرجات للحصول على ترتيب الطلاب حسب المجموع ومعرفة عدد الطلاب الناجحين والراسبين... الخ . ثم يتم إعلان النتيجة التي تضم أسماء الطلاب ومجموع درجاتهم وترتيبهم الخ .



□ يتضح لنا من هذا المثال ما يلي :

١. البيانات: أسماء وأرقام الطلاب ودرجاتهم في المواد المختلفة .
٢. المعلومات: النتائج التي تضم أسماء الطلاب ومجموع درجاتهم وترتيبهم .. الخ.
٣. معالجة البيانات: جمع الدرجات (عمليات حسابية) ، فرز الدرجات و ترتيب الطلاب (عمليات منطقية) .

- ملحوظة: في هذا المثال نجد أن عمليات المعالجة تمت يدوياً فلذلك نجد أنها تستغرق وقتاً طويلاً كما يحتمل وقوع أخطاء أثناء إجراء العمليات الحسابية ولكن باستخدام الحاسب يمكن تنفيذ كل الأعمال السابقة بطريقة آلية في وقت قليل وبدقة عالية جداً.



مصطلحات البرمجة

- **اللغة Language:** هي وسيلة تفاهم بين البشر من خلالها يتم التعبير عن حاجاتهم و متطلباتهم.
 - **لغة البرمجة Programming Language:** هي وسيلة تفاهم بين الإنسان و الحاسب يتم من خلالها كتابة الأوامر إلى الحاسب ليقوم بتنفيذها بدقة و سرعة عالية جداً.
 - **البرنامج Program:** هو مجموعة الأوامر و التعليمات المتسلسلة منطقياً لإيجاد حل مسألة ما، و المكتوبة بإحدى لغات البرمجة حيث يتم إدخال البرنامج إلى الحاسب فيقوم بتنفيذه.
 - **البرمجة Programming:** هي علم كتابة حلول المشاكل التي تواجهنا في الحياة بواسطة إحدى لغات البرمجة على شكل أوامر يقوم الحاسب بتنفيذها.
 - **البرمجيات Software:** هي مجموعة البرامج التي يكتبها المبرمجون من أجل الاستفادة القصوى من إمكانيات الحاسب الضخمة في مجالات الحياة المتنوعة.
 - **معالجة البيانات Data Processing:** هي إجراء العمليات الحسابية و المنطقية على البيانات Data لاستخلاص المعلومة المفيدة Information.
١. العمليات الحسابية Arithmetic Operations (+ ، - ، x ، /).
 ٢. العمليات المنطقية Logic Operations (> ، < ، = ، ≠ ، not).



خطوات كتابة البرامج و الخوارزميات

- قبل أن نتمكن من كتابة برنامج بلغة باسكال Pascal أو أي لغة أخرى لا بد لنا أولاً من فهم منطقي لجميع الخطوات الواجب إتباعها لحل مسألة ما بواسطة الحاسب.

١. الخطوة الأولى: تحديد و تعريف المشكلة

٢. الخطوة الثانية: تصميم البرنامج عن طريق كتابة خوارزمية حل المسألة Algorithm.

٣. الخطوة الثالثة: تحويل الخوارزمية إلى تعليمات بلغة باسكال (برنامج) أو أي لغة برمجة أخرى.

٤. الخطوة الرابعة: تحويل تعليمات لغة باسكال بدورها إلى لغة الآلة Binary Code المستعملة بواسطة برنامج المصنف Compiler و التأكد من خلّوها من كافة أنواع الأخطاء.

٥. الخطوة الخامسة: توثيق البرنامج

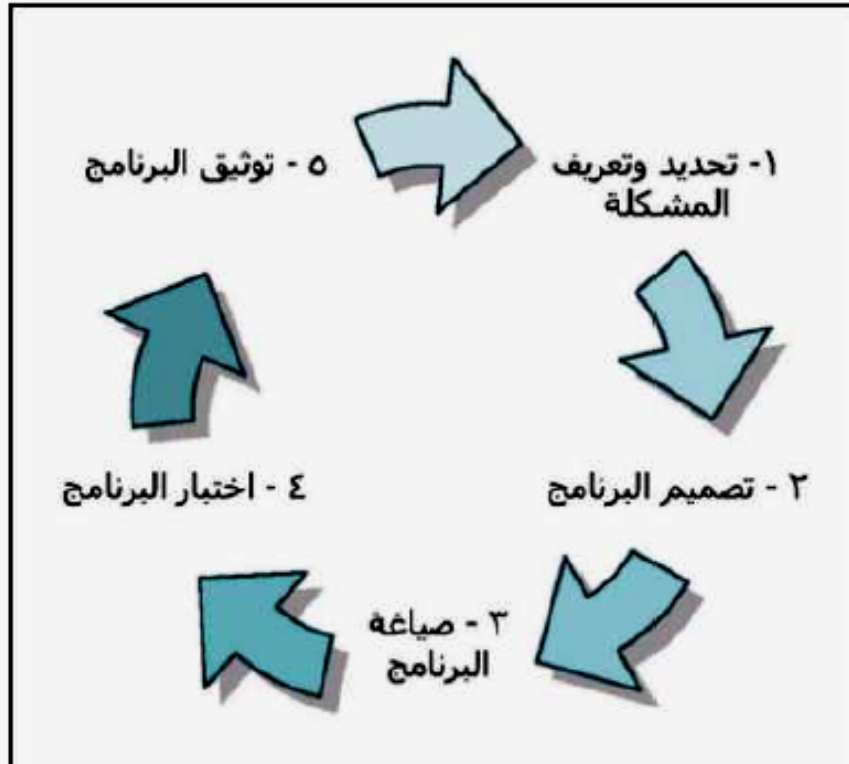
- بناءً عليه يجب التعرّف على مجموعة مفاهيم منها الخوارزمية و البرنامج و بعض المصطلحات و التسميات المستخدمة في برمجة الحواسيب.

خطوات صياغة وتطوير البرامج

Program Development Steps

١. تحديد وتعريف المشكلة Defining the Problem

- في هذه الخطوة يقوم المبرمج بتحديد وتعريف المشكلة وتتضمن هذه الخطوة تحديد التالي بالترتيب:
١. الهدف من البرنامج (حساب ارباح، فواتير استهلاك الماء والكهرباء، أو حساب معدل الطالب التراكمي)
 ٢. نوع وحجم المخرجات ووسائل الإخراج (تقارير - فواتير - شيكات - نقود ...)
 ٣. نوع وحجم البيانات المدخلة ووسائل الإدخال.
 ٤. مستخدمي البرامج والمستفيدين منه.



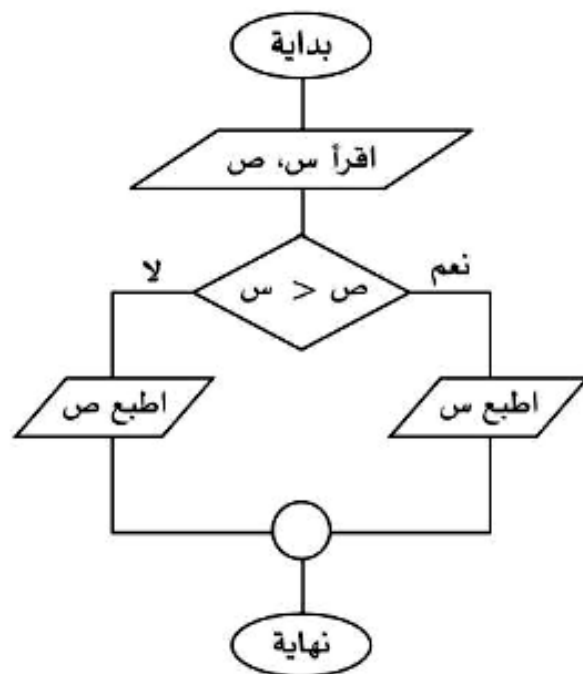
خطوات صياغة وتطوير البرامج

Program Development Steps

٢. تصميم البرنامج Design the Program

- يتم هنا تحديد المواصفات والخطوات الدقيقة والمرتبة منطقياً والتي تم فهمها ودراستها في الخطوة الأولى.
- ويتم ذلك باستخدام عدة طرق منها خرائط التدفق **Flowchart** ويطلق عليها أيضاً خرائط سير العمليات وهي مجموعة من الرموز المتعارف عليها تستخدم لتوضيح الخطوات المنطقية اللازمة لحل مشكلة ما.

أهم الرموز المستخدمة في خرائط التدفق



الرمز	الاسم
	بداية - نهاية Start - End
	مدخلات - مخرجات Input - Output
	معالجة Processing
	قرار Decision

خطوات صياغة وتطوير البرامج Program Development Steps

٣. صياغة البرنامج Coding the Program

- بعد الانتهاء من تصميم البرنامج يتم اختيار إحدى لغات البرمجة المناسبة لصياغة أوامر البرنامج Coding وذلك بالاستعانة بخريطة التدفق Flow Chart أو غيرها.
- يجب عند صياغة البرنامج اتباع قواعد صيانة لغة البرمجة المستخدمة حيث ان لكل لغة برمجة قواعد خاصة بها ولا يعمل البرنامج اذا كان هنالك اخطاء املائية او اخطاء في قواعد اللغة Syntax Errors.

خطوات صياغة وتطوير البرامج Program Development Steps

٤. اختبار البرنامج وتصحيح الأخطاء Program Debugging and Testing

■ يسمى البرنامج بعد صياغته باحدى لغات البرمجة البرنامج المصدر **Source Program** ولا يتم تنفيذه مباشرة على الحاسوب بل يتم ترجمته الى برنامج مكتوب بلغة الآلة **Object Program**.

■ تسمى عملية تحويل البرنامج المصدر الى برنامج الهدف **بالترجمة Compilation** ويقوم بها برنامج يسمى **المترجم Compiler**.

■ خلال عملية الترجمة **Compilation** قد تظهر أخطاء في صياغة البرنامج المصدر ينبغي



خطوات صياغة وتطوير البرامج

Program Development Steps

■ هناك ثلاث انواع من الأخطاء:

١. أخطاء في قواعد اللغة **Syntax Errors**: اخطاء املائية في كتابة الأوامر.

٢. أخطاء منطقية **Logical Errors**: لا يكتشفها الحاسوب وتظهر عند تنفيذ البرنامج على عينه من البيانات فنحصل على نتائج خاطئة او غير متوقعة، ويقوم المبرمج بتتبع خطوات البرنامج لمعرفة مصدر الخطأ وتصحيحه وتسمى هذه العملية **Tracing**.

٣. أخطاء اثناء التشغيل **Run-Time Errors**: تظهر عند تنفيذ البرنامج مثل عدم حجز مساحة كافية للمدخلات او الدخول في دوران بلا نهاية، وتظهر رسالة بنوع الخطاء.

٥. توثيق البرنامج **Documenting the Program**

■ في هذه المرحلة تتم كتابة وصف تفصيلي لصياغة البرنامج، ويشمل هذا التوثيق أصل المشكلة وخطوات الحل وخرائط الحل وتعليمات التشغيل ومتطلبات التشغيل والمدخلات والمخرجات وكيفية التحكم في البرنامج في المواقف المختلفة.

الخوارزمية Algorithm

- **الخوارزمية:** هي مجموعة الخطوات المتسلسلة منطقياً لحل مسألة ما و تكون مكتوبة على شكل خطة يدوية حيث يتم تنفيذ أمر واحد أو عدة أوامر في كل خطوة، و الهدف منها تبسيط حل المسألة وتقسيمها إلى خطوات مبسطة يسهل فهمها.
- سميت الخوارزمية بهذا الاسم نسبة إلى العالم العربي أبو جعفر محمد بن موسى الخوارزمي الذي ابتكرها في القرن التاسع الميلادي.
- **الخوارزمية (تعريف ٢):** مجموعة منتهية من التعليمات والتي بإتباعها يتم إنجاز مهمة محددة. هذه التعليمات يجب أن تكون محددة و خالية من الغموض كما يجب أن تكون بسيطة يمكن تطبيقها (من حيث المبدأ) من قبل أي شخص باستخدام الورقة و القلم فقط.
- تبحث الخوارزميات عن حلول لمسائل معينة في المعلوماتية بغية تحويلها إلى برامج ، كما تبحث أيضا في إيجاد الحل الأفضل في حال تعدد الحلول.
- **شكل الخوارزمية:**

- | | |
|--------------------|--|
| - Start | - البداية |
| - Enter first num | - التعريف بمعطيات المسألة |
| - Enter second num | - إدخال قيم معطيات المسألة |
| - Calc the sum | - معالجة المعطيات وفق القوانين (الرياضية و المنطقية) |
| - Print the result | - إخراج النتائج |
| - end | - النهاية |

بنية الخوارزمية Algorithm Structure

- هناك ثلاث تراكيب لبناء البرامج و كتابة الخوارزميات. الفكرة تكمن في أن أي برنامج أو خوارزمية يجب أن تتكون من هذه التراكيب الثلاثة فقط:

١- **التسلسل (Sequence):** تكون الخوارزمية عبارة عن مجموعة من التعليمات المتسلسلة، هذه التعليمات قد تكون إما بسيطة أو من النوعين التاليين.

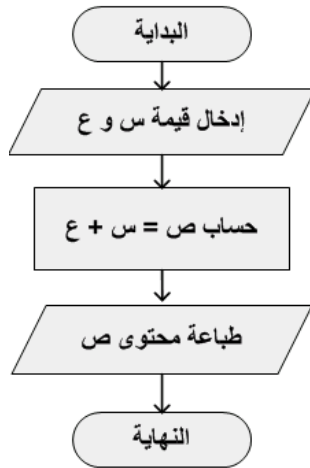
٢- **الاختيار (Selection):** بعض المشاكل لا يمكن حلها بتسلسل بسيط للتعليمات ، و قد تحتاج إلى اختبار بعض الشروط و تنظر إلى نتيجة الاختبار، إذا كانت النتيجة صحيحة تتبع مسار يحوي تعليمات متسلسلة و إذا كانت خاطئة تتبع مسار آخر مختلف من التعليمات. هذه الطريقة هي ما تسمى اتخاذ القرار أو الاختيار.

٣- **التكرار (Looping):** عند حل بعض المشاكل لا بد من إعادة تنفيذ نفس التعليمات وفق تسلسلها لعدد معين من المرات. و هذا ما يطلق عليه التكرار.

- و قد أثبت أنه لا حاجة إلى تراكيب إضافية. فاستخدام هذه التراكيب الثلاث يسهل فهم الخوارزمية و اكتشاف الأخطاء الواردة فيها و تغييرها.

تمثيل الخوارزمية / الشيفرة المزيفة

١- الشيفرة المزيفة (Pseudo code) : تمثيل الخوارزمية بلغات البشر كالإنكليزية أو الفرنسية أو العربية أو بلغات البرمجة كالباسكال Pascal البعض يستخدم الكثير من التفاصيل و البعض الآخر يستخدم القليل ... فلا قاعدة معينة لكتابة هذا النوع من الشيفرات.



• **مثال (١):** لكتابة برنامج يقوم بجمع عددين، نكتب الخوارزمية التالية :

١- بداية البرنامج

٢- قراءة العددين المدخل س و ع

٣- حساب ناتج جمع ص = س + ع

٤- اطبع ناتج الجمع ص

٥- نهاية البرنامج

• **مثال (٢):** لكتابة برنامج يحدد هل العدد المدخل زوجي ام فردي، نكتب الخوارزمية التالية :

١- بداية البرنامج

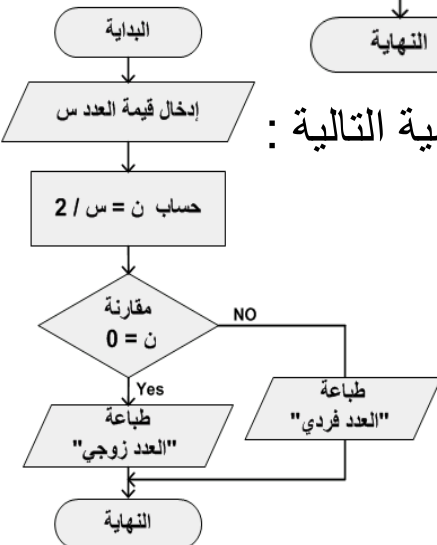
٢- قراءة العدد المدخل س

٣- حساب "ن" باقي قسمة العدد "س / ٢"

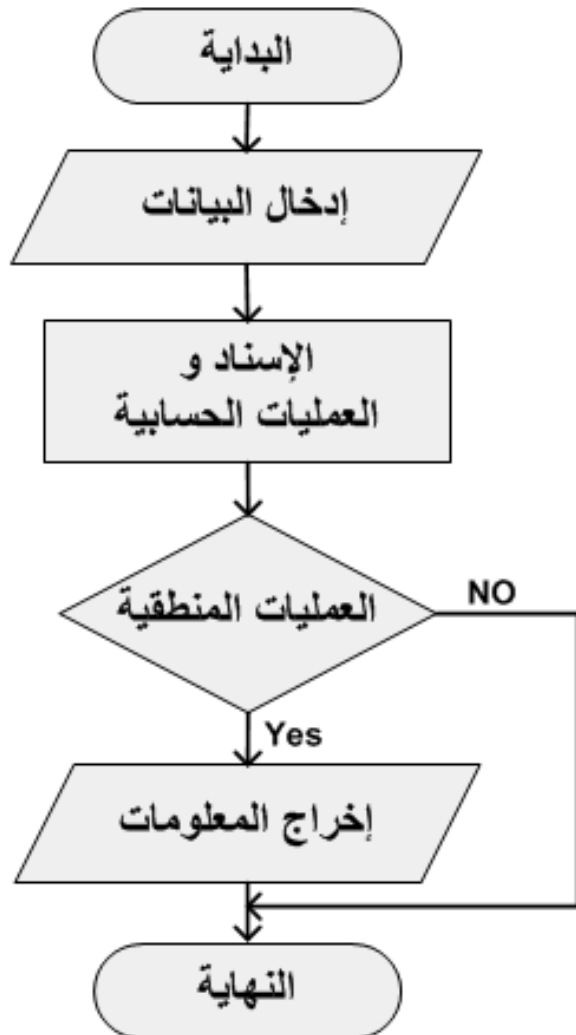
٤- مقارنة إذا كان ن = ٠ اطبع العدد زوجي

٥- و إلا اطبع العدد فردي

٦- نهاية البرنامج



تمثيل الخوارزمية / المخطط التدفقي

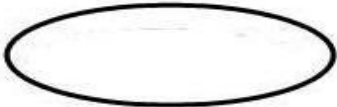
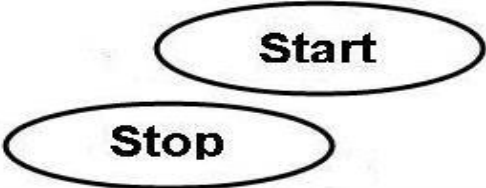
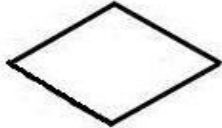
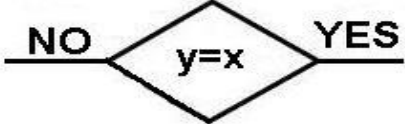
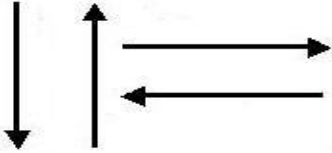
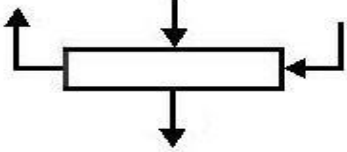
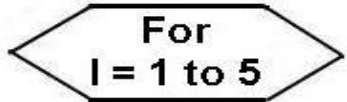


٢- المخطط التدفقي (Flowchart): هو وسيلة للتعبير عن

فكرة حل أي مسألة بواسطة أشكال هندسية محددة توضح تسلسل خطوات الحل.

- تعمد الطريقة البيانية لصياغة الخوارزميات على توضيح خطوات تنفيذ الخوارزمية باستخدام أشكال هندسية خاصة وأسهم تصل بينها إضافة إلى عبارات باللغة الطبيعية وتعابير رياضية أو منطقية. وبذلك نحصل على ما يسمى بالمخطط التدفقي Flowchart Diagram ونلاحظ هنا أن الأسهم تفصل العمليات اللازمة لإنجاز العمل و تبين تسلسلها يستخدم المخطط التدفقي أشكالاً هندسية متفق عليها خصص كل منها لنوع من العمليات.

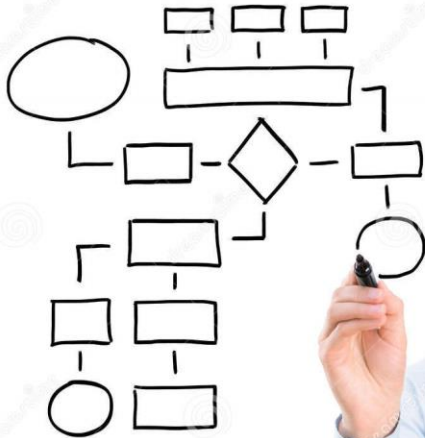
رموز المخطط التدفقي Flowchart Components

الرمز	الحدث الذي يمثله	مثال
	حدث طرفي Terminal لبيان بدء (Start) أو انتهاء (Stop) خريطة سير العمليات	
	عملية حسابية (Process)	
	إدخال / إخراج INPUT \ OUTPUT ادخال البيانات / اخراج معطومات من وإلى الحاسب	
	اتخاذ قرار Decision	
	اتجاه تتابع العمليات Flow Line	
	تكرار أو دوران Loop	

المخطط التدفقي Flowchart

• تُستخدم أشكال هندسية خاصة في الرسم للتعبير عن عمليات برمجية محددة.

• المخطط يبدأ من الأعلى ثم ينساب و ينزل للأسفل، وهذه الأشكال المستخدمة للتعبير عن عمليات البرمجة :



• أنواع المخططات التدفقية:

١- المخطط التدفقي تتابعي.

٢- المخطط التدفقي المتفرع.

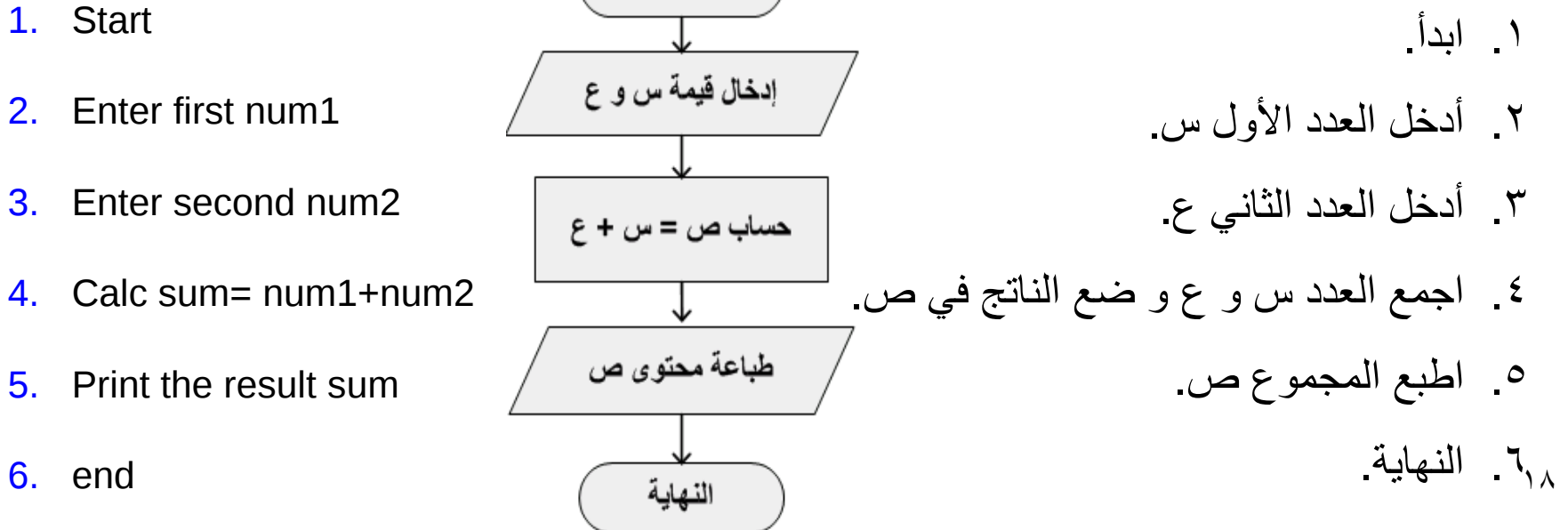
٣- المخطط التدفقي ذو الدورانات المتكررة.

• إن اختيار المخطط يعتمد على نوع المسألة التي نقوم بحلها. و لن نتضح طريقة استخدام هذه الأشكال في بناء المخطط التدفقي إلا بعد عرض عدد من الأمثلة .

المخطط التدفقي التتابعي

- إن أبسط أنواع البرامج هو البرنامج التي تأتي أفكاره و خطوات حله متسلسلة خلف بعضها البعض، أي خطوة بعد خطوة بحيث أنك عندما تقرأها تجدها فكرة واحدة مترابطة.
- لتمثيل هذا النوع من البرامج فإننا نرسم مخطط تدفقي متتالي لا يوجد به أي تفرعات. و لكي تتضح الفكرة أكثر و أكثر فسوف نقوم برسم المخطط التدفقي لبرنامج يقرأ عددين ثم يقوم بطباعة ناتج الجمع، قبل رسم المخطط التدفقي أقترح أن نقوم بكتابة الخوارزمية كنوع من التمرين .

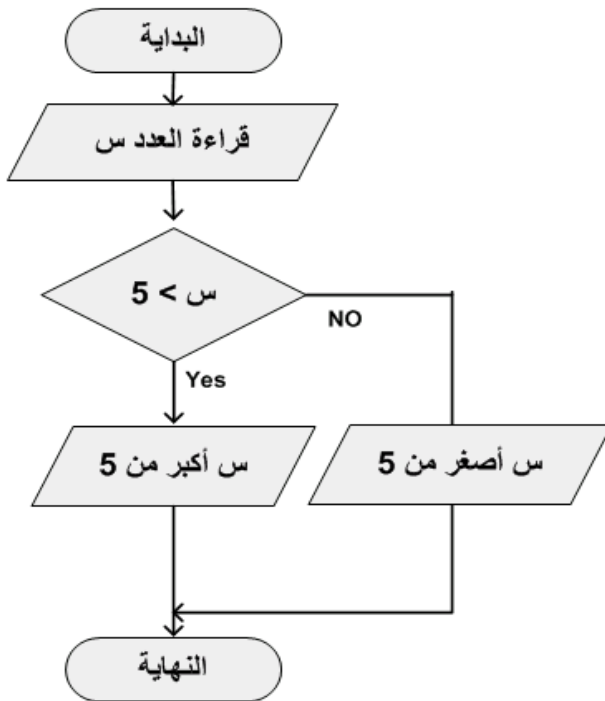
شفيرة الخوارزمية المزيفة Pseudo Code:



المخطط التدفقي المتفرع

- هناك عدد من البرامج فيها درجة بسيطة من التعقيد، و ذلك لاحتواء البرنامج على عدد من الشروط التي تستوجب عليك تنفيذ أمر معين إذا كان الشرط صحيحاً.
- أما إذا كان الشرط خاطئاً فإن البرنامج سيتفرع لينفذ أمر آخر.
- من هذه البرامج ما تقوم بتحديد إذا كان الطالب ناجحاً أم لا، أو بتحديد إذا كان الرقم سالب أم موجب. و سيتم توضيح كيفية المخطط التدفقي المتفرع من المثال التالي حيث يقوم البرنامج بقراءة عدد ثم يطبع رسالة تُبين هل العدد أكبر

• شيفرة الخوارزمية المزيفة Pseudo code:

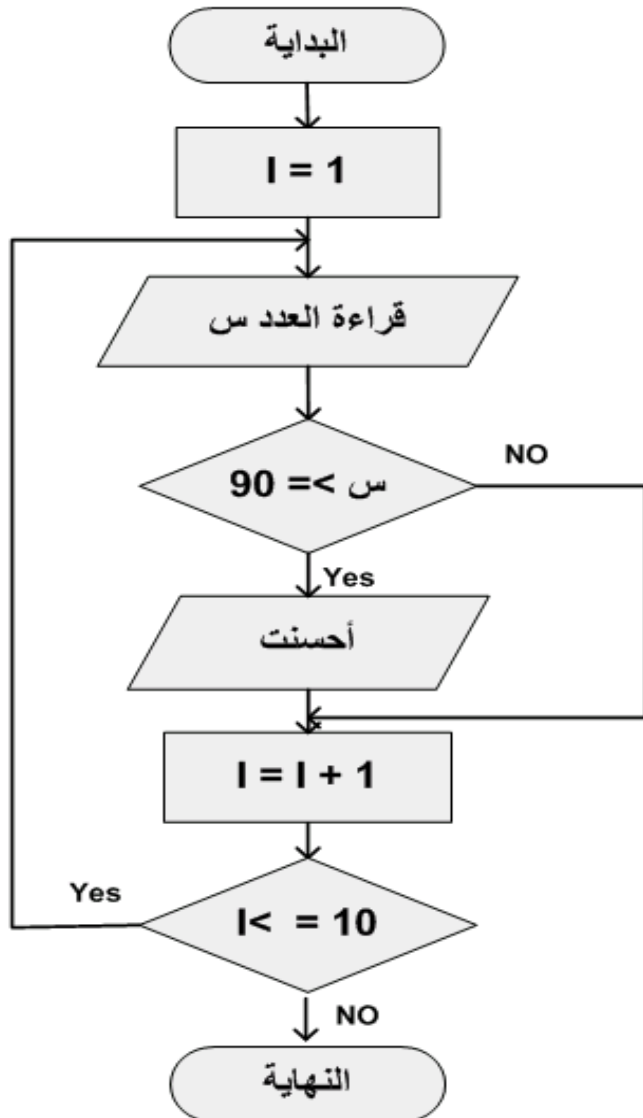


١. إبداء
٢. أدخل العدد س.
٣. هل العدد (س) أكبر من خمسة
٤. إذا كانت النتيجة نعم، اطبع "العدد أكبر من خمسة."
٥. إذا كانت النتيجة لا، اطبع "العدد أصغر من خمسة."
٦. توقف.

المخطط التدفقي ذو الدورانات المتكررة

- بعض البرامج التي تكون في أحد مراحل التنفيذ تستلزم أن يقوم البرنامج بتكرار مجموعة من عمليات البرنامج أكثر من مرة.
- مثال: برنامج يقوم بطباعة عبارة "أحسننت" إذا حصل الطالب على علامة أكبر أو يساوي ٩٠ في عشرة مقررات دراسية.

1. Start
2. Set the counter $I = 1$
3. Input X
4. If $X \geq 90$ then Print "**Excellent**" Else Continue
5. Add $I = I + 1$
6. If $I \leq 10$ then Go to Step 3
7. end



أمثلة على الخوارزميات

Algorithm Examples

مثال ١: اكتب خوارزمية إيجاد متوسط درجات طالب في ثلاث مقررات دراسية و
ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- المخرجات : طباعة متوسط الدرجات ، ونرمز له بـ (Av)
- ب- المدخلات : درجات الطالب الثلاث ونرمز لهم بـ $(M1, M2, M3)$
- ت- العمليات : حساب متوسط الدرجات $Av = (M1+M2+M3) / 3$

٢. كتابة خوارزمية الحل (الشفرة المزيفة):

- ١. البداية
- ٢. أدخل الدرجات الثلاث $M1, M2, M3$
- ٣. احسب متوسط الدرجات الثلاث من المعادلة : $Av = (M1+M2+M3) / 3$
- ٤. اطبع متوسط درجات الطالب Av
- ٥. النهاية

التوضيح	مخطط الانسياب
بداية المخطط التدفقي	START
شكل يوضح أن العملية المراد تنفيذها عبارة عن إدخال قيم درجات المواد الثلاثة	M1,M2,M3
شكل يوضح عملية حسابية وهي عبارة عن حساب المتوسط لدرجات المواد الثلاثة	$Av = (M1+M2+M3) / 3$
شكل يوضح أن العملية ما هي إلا إخراج قيمة المتوسط	Av
نهاية المخطط التدفقي	END

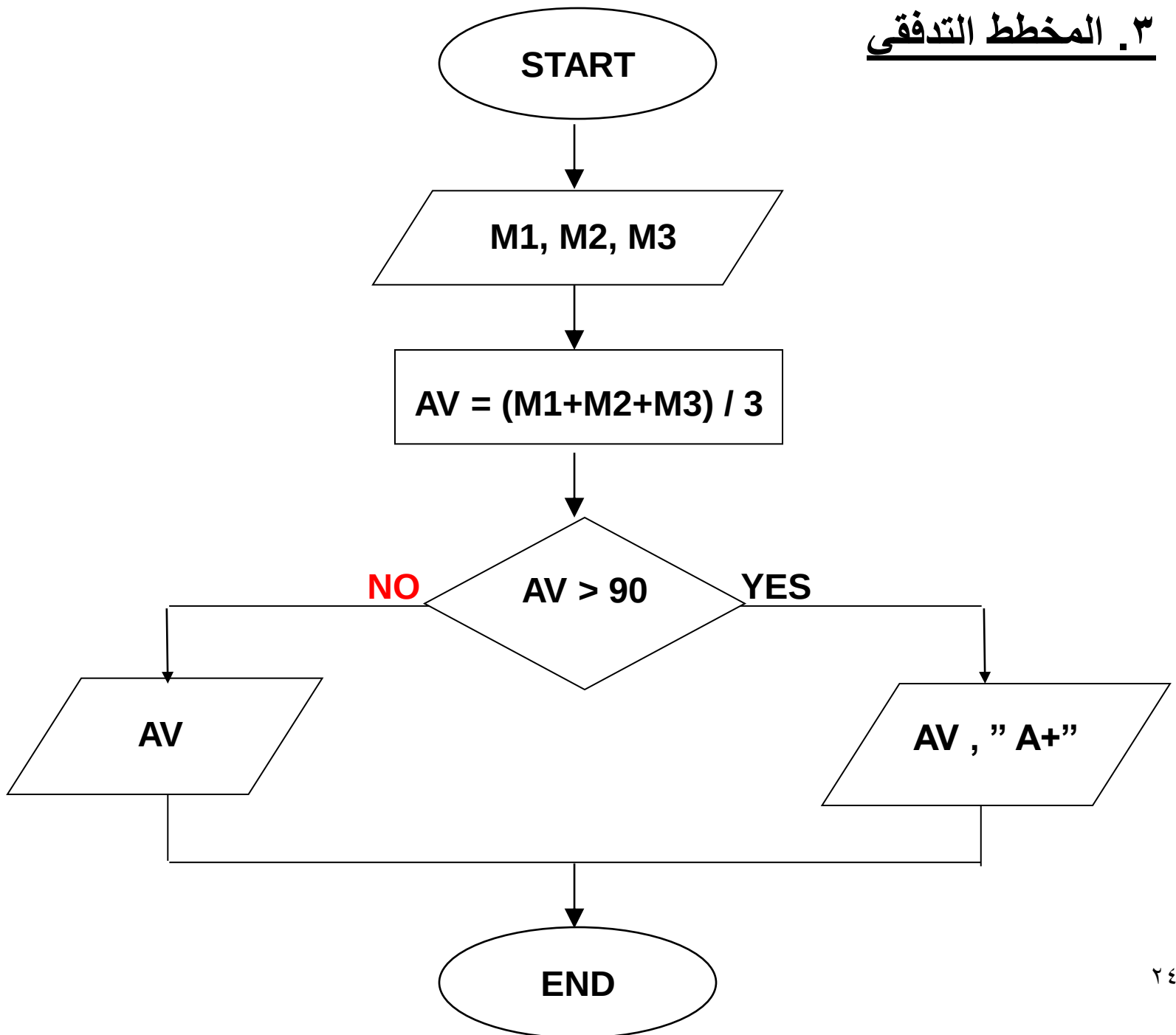
مثال ٢: اكتب خوارزمية إيجاد متوسط درجات طالب في ثلاث مقررات دراسية و اطبع عبارة "ممتاز" إذا كان متوسط الدرجات أكبر من ٩٠ و إلا فاطبع المتوسط فقط و ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- **المخرجات :** متوسط الدرجات ، ونرمز له بـ (Av)
- ب- **المدخلات :** الدرجات الثلاث ونرمز لهم بـ $(M1, M2, M3)$
- ت- **العمليات :** قانون متوسط الدرجات $Av=(M1+M2+M3)/3$

٢. كتابة خوارزمية الحل (الشفرة المزيفة):

- ١. البداية
- ٢. أدخل الدرجات الثلاث $M1, M2, M3$
- ٣. احسب متوسط الدرجات الثلاث من المعادلة : $Av=(M1+M2+M3)/3$
- ٤. إذا كان المتوسط أكبر من ٩٠ فإذهب إلى الخطوة ٥
و إلا فإذهب إلى الخطوة ٦
- ٥. اطبع نتيجة المتوسط Av والتقدير ممتاز "A+" .. اذهب إلى الخطوة ٧
- ٦. اطبع نتيجة المتوسط Av
- ٧. النهاية



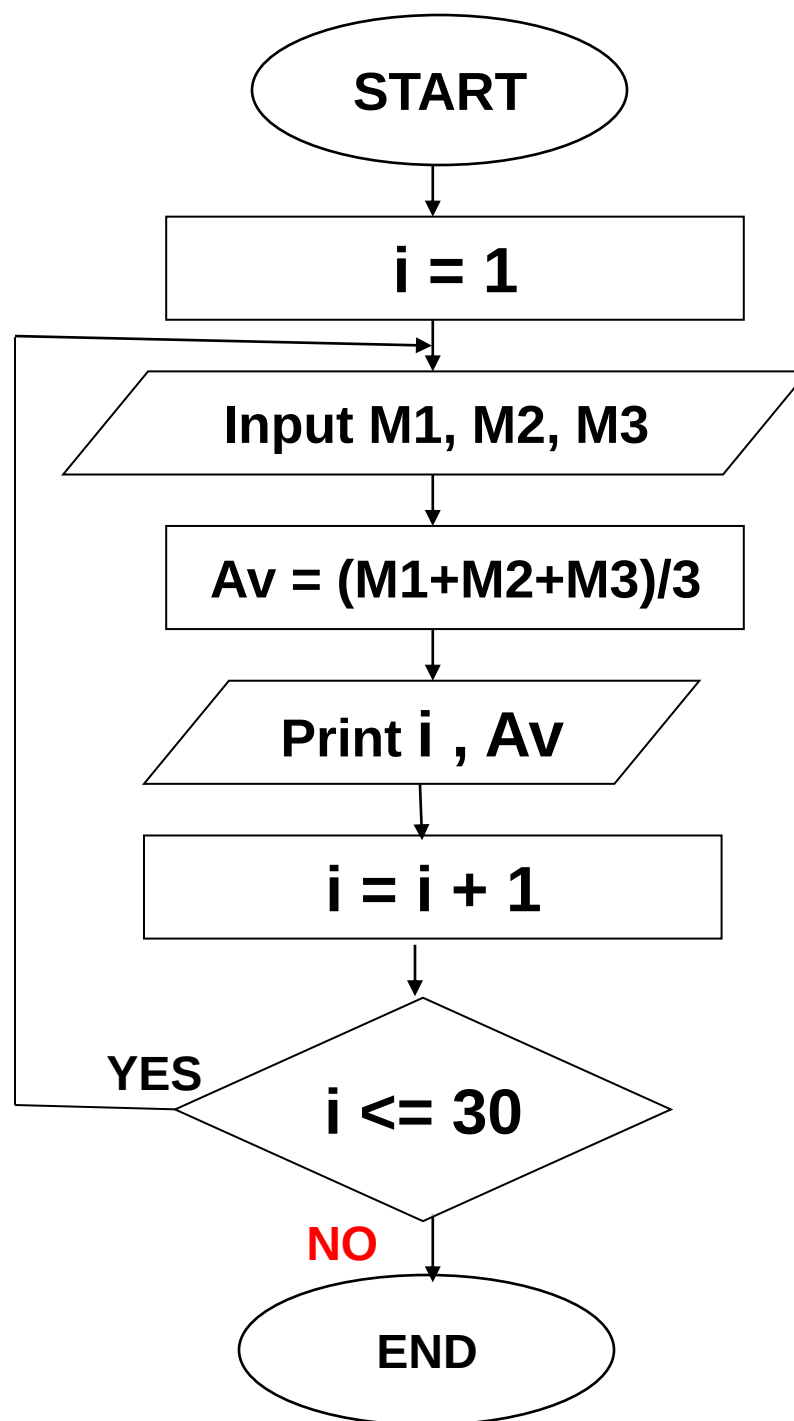
مثال ٣: اكتب خوارزمية لقراءة درجات ٣٠ طالب في ثلاث مقررات دراسية و اطبع رقم الطالب و متوسط درجاته و ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- المخرجات : أرقام ٣٠ طالب ومتوسط درجاتهم (Av)
- ب- المدخلات : رقم الطالب (i) ، ودرجاته الثلاث ($M1, M2, M3$)
- ت- العمليات : وضع قيمة ابتدائية للعداد $i = 1$
ثم حساب متوسط الدرجات $Av = (M1 + M2 + M3) / 3$
ثم إضافة واحد للعداد $i = i + 1$

٢. كتابة خوارزمية الحل (الشفيرة المزيقة):

١. البداية
٢. اجعل العداد يساوي واحد $i = 1$
٣. أدخل درجات الطالب رقم (i) الثلاث ($M1, M2, M3$)
٤. احسب متوسط الدرجات الثلاث $Av = (M1 + M2 + M3) / 3$
٥. اطبع رقم الطالب (i) ، ومتوسط الدرجات (Av)
٦. أضف واحد للعداد $i = i + 1$
٧. إذا كان العداد ($i \leq 30$) أقل من أو يساوي ٣٠ فاذهب إلى الخطوة ٣
٨. النهاية.



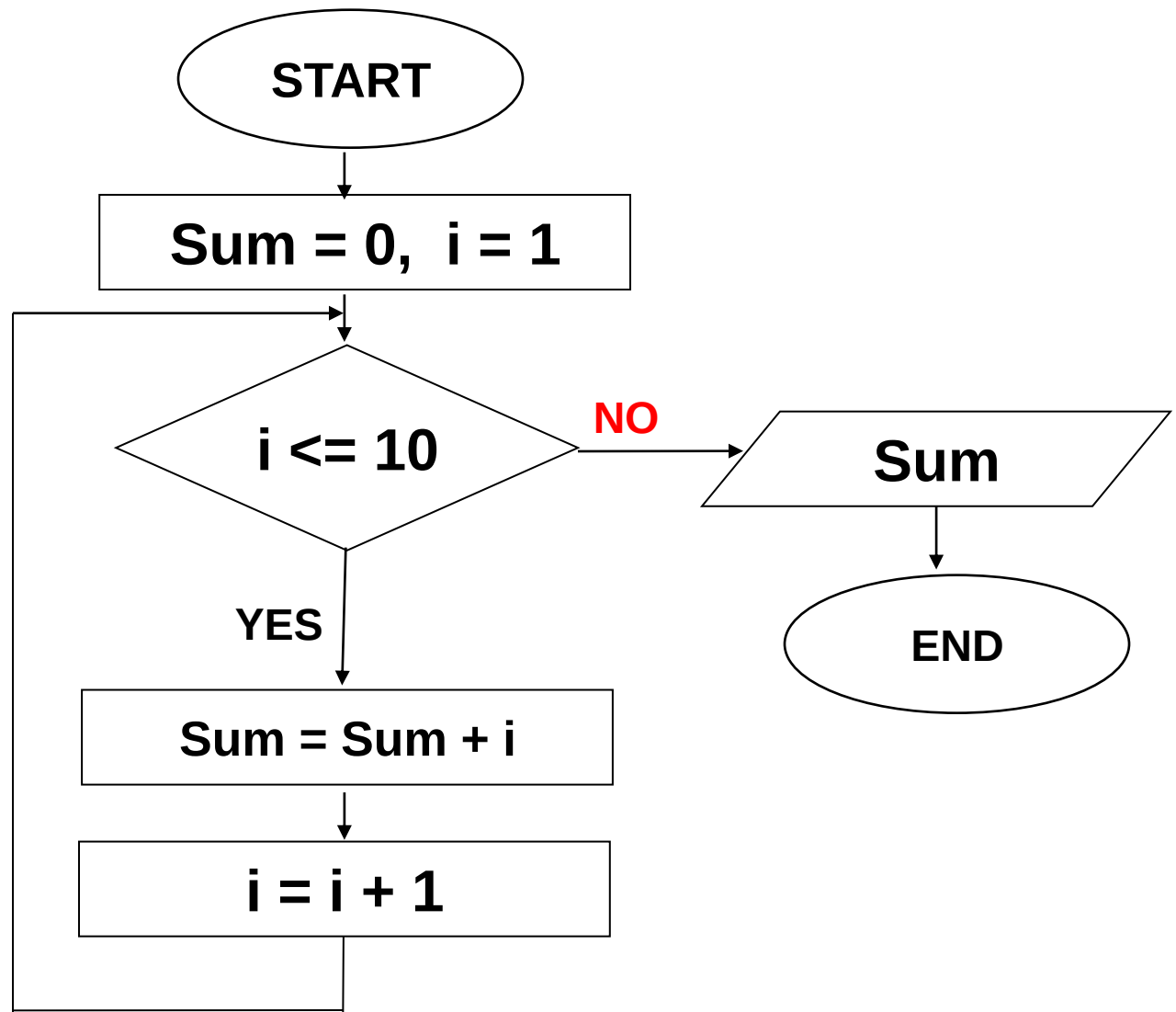
مثال ٤: اكتب خوارزمية لحساب و طباعة مجموع الأعداد من ١ إلى ١٠ و ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- المخرجات : طباعة مجموع الأعداد من ١ إلى ١٠ (Sum)
- ب- المدخلات : الأعداد من ١ إلى ١٠ من خلال عداد نرسم له بالرمز (i)
- ت- العمليات : إذا كان $i \leq 10$ اطبع Sum
 $Sum = Sum + i$
ثم أضف واحد للعداد $i = i + 1$ وابدأ من جديد

٢. كتابة خوارزمية الحل (الشفيرة المزيقة):

١. البداية
٢. ضع قيمة ابتدائية للمجموع تساوي صفر $Sum = 0$ وقيمة للعداد تساوي الواحد $i=1$
٣. هل العداد أقل من أو يساوي ١٠
إذا كان نعم فاذهب إلى الخطوة ٤
إذا كان لا فاذهب إلى الخطوة ٥
٤. احسب المجموع $Sum = Sum + i$
٥. اطبع المجموع (Sum)
٦. النهاية



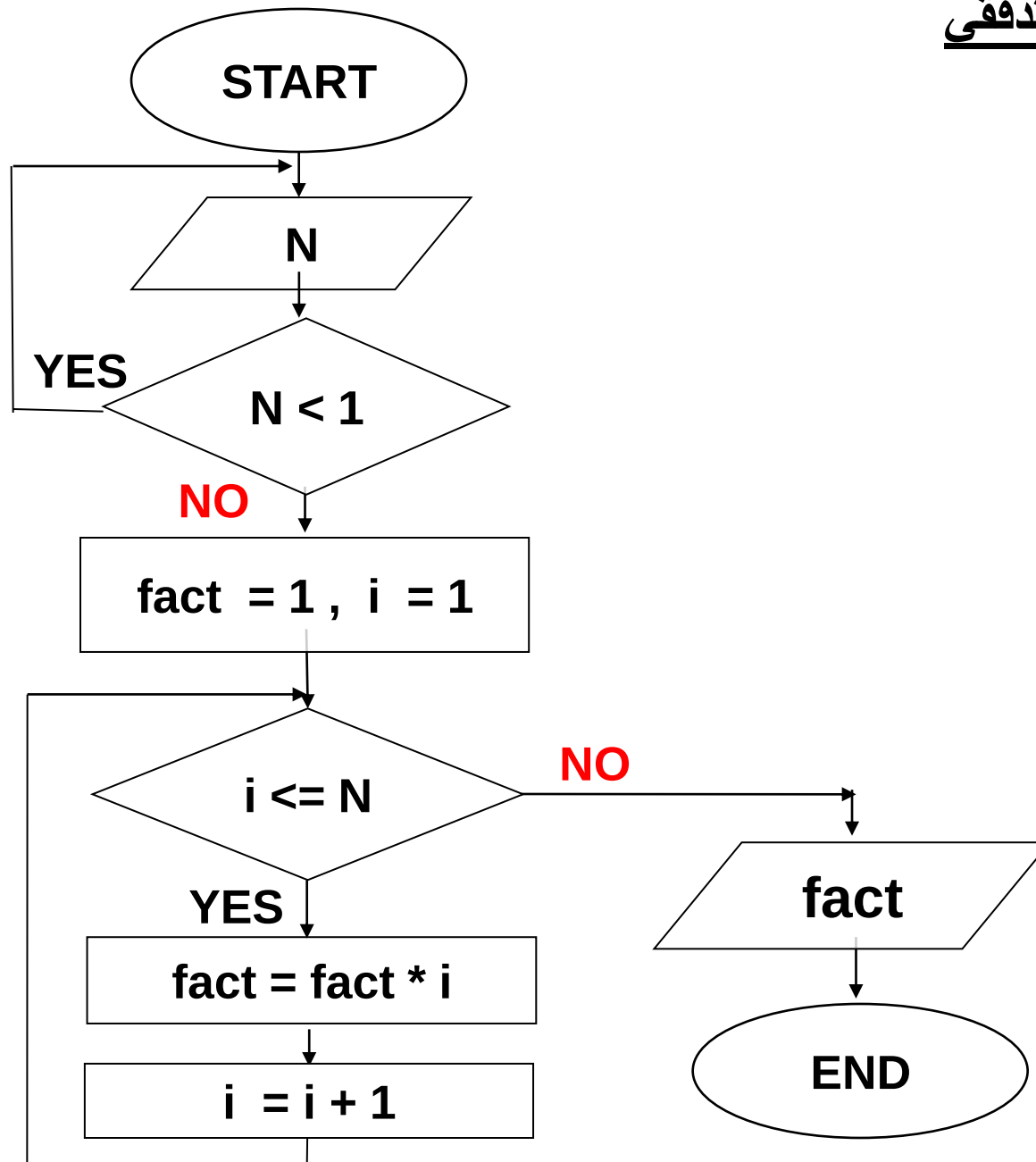
مثال ٥: اكتب خوارزمية لحساب و طباعة قيمة العدد $N!$ عاملي (فاكتوريال) و ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- المخرجات : قيمة مضروب العدد $N!$ (Factorial)
- ب- المدخلات : قيمة العدد (N)
- ت- العمليات : قانون المضروب $\text{Factorial} = N * \text{fact}(N-1)$

٢. كتابة خوارزمية الحل (الشفيرة المزيفة):

١. البداية
٢. أدخل قيمة العدد N
٣. إذا كان $N < 1$ فاذهب إلى الخطوة ٢ و إلا تابع ..
٤. ضع قيمة ابتدائية للناتج $\text{fact} = 1$ و قيمة للعداد $i = 1$
٥. إذا كان العداد $N \leq i$ فاذهب إلى الخطوة 6 و إلا فاذهب إلى الخطوة ٨
٦. حساب مضروب العدد $\text{fact} := \text{fact} * i$
٧. أضف واحد للعداد $i := i + 1$ و اذهب إلى الخطوة ٥
٨. اطبع قيمة fact
٩. النهاية



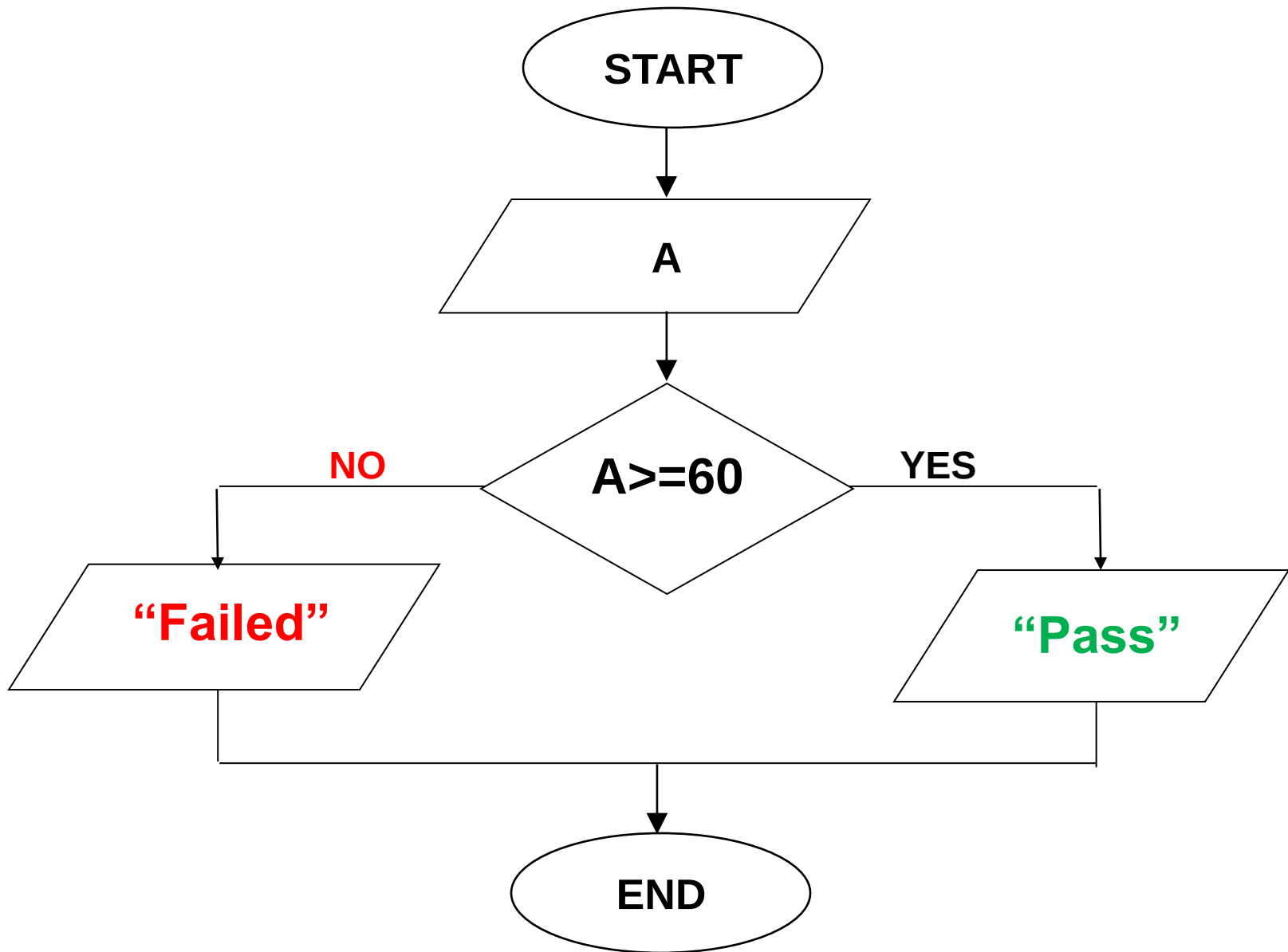
مثال ٦: اكتب خوارزمية لطباعة كلمة ناجح Pass إذا كانت علامة الطالب أكبر أو يساوي ٦٠ أو راسب Failed و ارسم المخطط التدفقي لخوارزمية الحل

١. تحديد عناصر المسألة

- أ- المخرجات : طباعة عبارة "Pass" أو "Failed"
- ب- المدخلات : علامة الطالب ونرمز له بالرمز (A)
- ت- العمليات : إذا كانت علامة الطالب $A \geq 60$ اطبع Pass و إلا فاطبع Failed

٢. كتابة خوارزمية الحل (الشفيرة المزيفة):

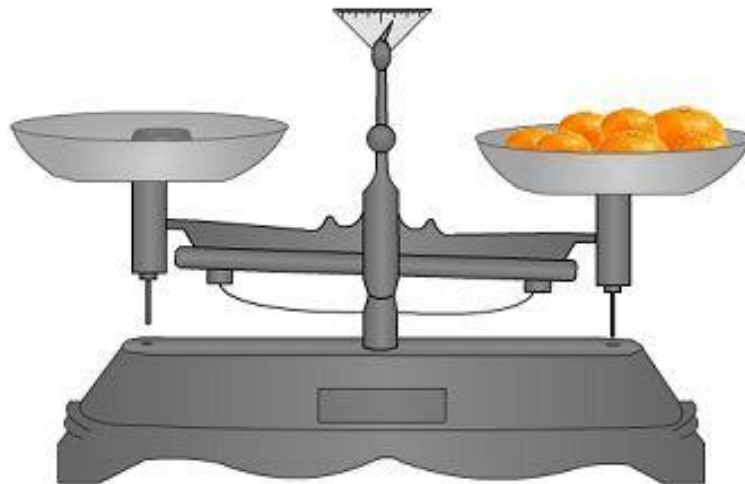
- ١. البداية
- ٢. أدخل علامة الطالب A
- ٣. إذا كانت علامة الطالب $A \geq 60$ انتقل إلى الخطوة ٤
و إلا انتقل إلى الخطوة ٥
- ٤. اطبع عبارة "Pass"
- ٥. اطبع عبارة "Failed"
- ٦. النهاية



تقييم الخوارزمية Algorithm

- بعد الانتهاء من كتابة خوارزمية حل مسألة معينة والتحقق من صحتها، تأتي الخطوة اللاحقة وهي **تقييم** أداء هذه الخوارزمية بشكل عام، و تقييم أداء الخوارزمية هو دراسة تعقيد الخوارزمية أي الزمن اللازم لتنفيذها. و يمكن إيجاد الزمن اللازم لتنفيذ خوارزمية معينة بطريقتين:

١. **التحليل التجريبي:** تشغيل عداد زمني لقياس الزمن المستغرق
٢. **التحليل النظري:** يُعبّر عن زمن تنفيذ خوارزمية كدالة لحجم الدخل فيرمز بـ $T(n)$ للزمن المستغرق لتنفيذ خوارزمية معينة بالنسبة لأي دخل من الحجم n .

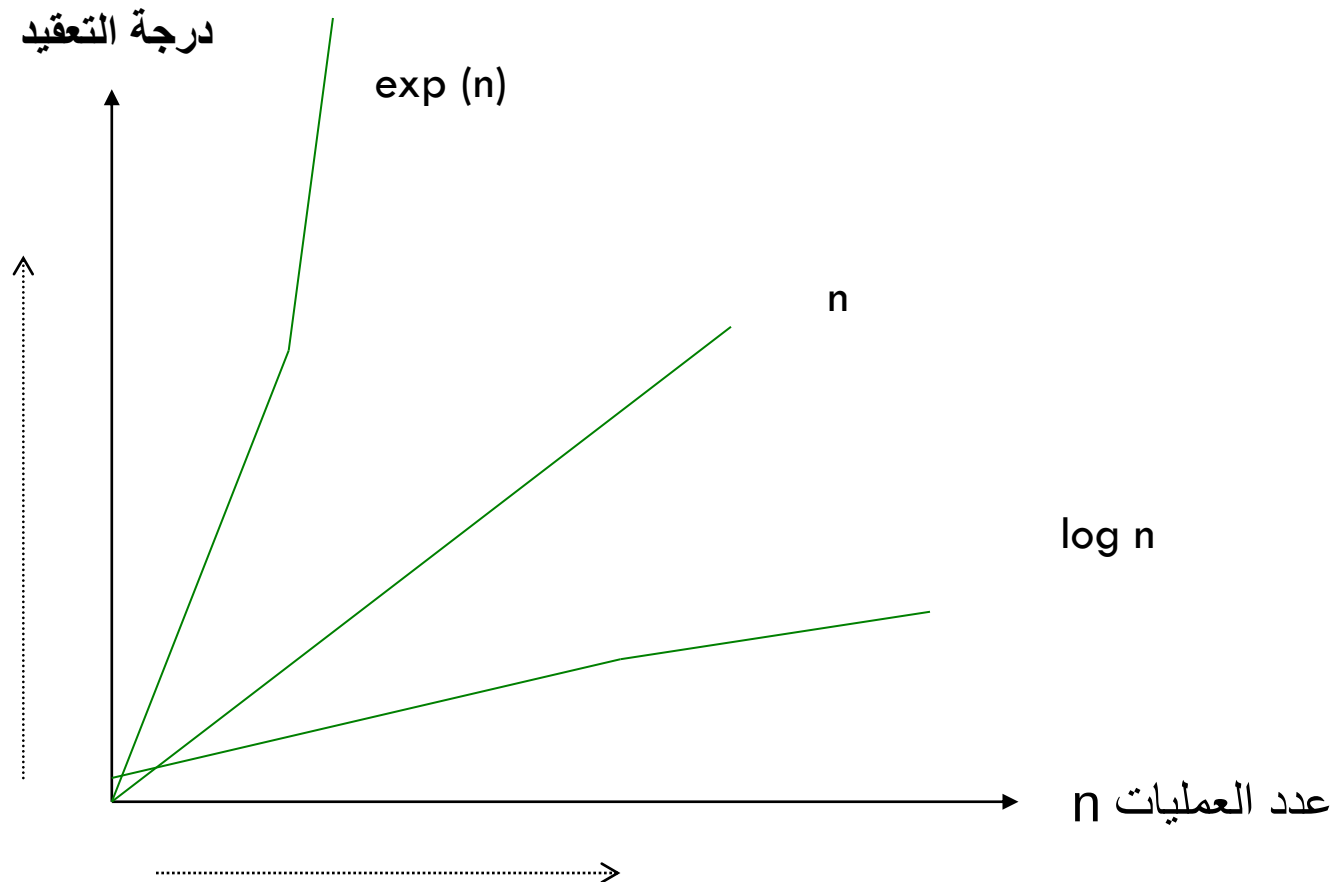


تعقيد الخوارزمية Complexity of an Algorithm

- بشكل عام من أجل حل مشكلة ما، بإمكاننا أن نجد أكثر من خوارزمية واحدة للحل المنشود، و السؤال الذي يطرح نفسه ما الذي يجعل خوارزمية أفضل من الأخرى؟ ماذا نعني بخوارزمية فعّالة Efficient Algorithm؟
 - في الواقع، السؤال هو لم نحن بحاجة لإيجاد أفضل وأكثر الخوارزميات فعاليةً بما أن الحاسبات تصبح أسرع فأسرع ؟
 - (ينص قانون مور Moore's low على أن عدد الترانزستورات في وحد المعالجة المركزية CPU يتضاعف كل ١٨ شهراً، أي أنّ أداء الكمبيوتر يتضاعف أيضاً.
 - يقصد بمفهوم دراسة الخوارزميات هو دراسة ما تحتاجه هذه الخوارزميات من وقت و مساحة تخزين للوصول للحل، و يعرف هذا العلم باسم Complexity of Algorithms
١. الوقت (يعتمد على عدد خطوات)
 ٢. مساحة تخزين (يعتمد على عدد المعلومات التي نحتاج لتخزينها في الذاكرة العشوائية)
- فمثلاً: إذا أردنا تخزين عنصر جديد في المصفوفة، فنحن نحتاج لحساب عدد العناصر التي نحتاج لتحريكها بالإضافة لعملية التخزين، لنحسب الوقت و المساحة.
- هناك مقادير عامة لمدى تعقيد الوقت المستخدم، و هي موضحة بالشكل التالي:

مخطط يوضح درجة تعقيد الخوارزمية

Complexity of an Algorithm



تعقيد الخوارزمية Complexity of an Algorithm

- و يعرف Time complexity بأنه الوقت اللازم لتنفيذ البرنامج (الخوارزمية) مقارنة بعدد المدخلات (مثلا عدد عناصر المصفوفة).
- و يعرف Space complexity بأنها مساحة التخزين اللازمة لتنفيذ البرنامج مقارنة بعدد المدخلات.
- و طالما أنها مرتبطة بالمدخلات، فستجد أنّ درجة تعقيد الخوارزمية تقاس بما يعرف بـ $O(n)$ أو Big o notation علما بأنّ n هنا يقصد بها عدد مرات المعالجة أو التخزين، و كلما قلة قيمة n كلما كانت هذه الخوارزمية أفضل من حيث الوقت و التخزين.
- نلاحظ أيضاً أن تحقيق أحد الشرطين في مسألة ما ينفي الشرط الآخر. إذا كان البرنامج سينفذ عدداً قليلاً من المرات فالنقطة الأولى أهم، وإذا كان سينفذ عدداً كبيراً من المرات فالنقطة الثانية أهم، فالمسألة أصبحت متعلقة بالمستخدم.

تعقيد الخوارزمية Complexity of an Algorithm

- النقطتان السابقتان (على الرغم من أهمية أخذهما بعين الاعتبار عند تطوير البرمجيات) لا يمكن اعتبارهما معياراً لفعالية الخوارزمية، ومن هنا نظهر أهمية فكرة التعقيد Complexity. حاول أن تنفذ الخوارزميتين التاليتين على جهازك:

• الأولى: $sum = n * (n + 1) / 2 ;$

• الثانية: $for\ i := 1\ to\ n\ do$

$sum := sum + i ;$

- إنّ كلا الخوارزميتين تقومان بجمع الأعداد من ١ إلى n . (تحليل رياضي ١). كلاهما سهل التنفيذ ولا يستهلك الكثير من موارد النظام.
- من أجل قيم صغيرة لـ n لا نلاحظ فرقاً في زمن التنفيذ (الفرق موجود لكنه غير محسوس).
- لكن إذا جعلنا قيمة $n = 10000000$ عندها سنلاحظ أن الثانية ستستهلك وقتاً أطول من الأولى مهما كان الجهاز أو لغة البرمجة المستخدمين.
- بعد أن ندرس التعقيد سنلاحظ أن الخوارزمية الأولى من أجل قيمة معينة لـ n تنفذ عمليات أقل من تلك التي تنفذها الثانية مما يجعل زمن تنفيذها أقل وبالتالي تعقيدها الزمني أصغر مما يجعلها أكثر فعالية. فالخوارزمية ذات التعقيد الزمني الأصغر هي الأكثر فعالية. وهذا هو المعيار الذي نبحث عنه.