بسم الله الرحمن الرحيم

جامعة النيل الأزرق

كلية الدراسات الإضافية

أساسيات البرمجة

fundamentals of programming

الجزء الأول

إعداد : أ. عبد الرحمن عباس إبراهيم

2007

مقدمة Introduction:

تتجه كل المؤسسات الحكومية منها و المدنية ، التجارية و العسكرية و المدارس ، حتى في المنازل في المطابخ و الغرف ، تتجه لاستخدام الحاسب الآلي في تنفيذ الأعمال ، وهذا يتطلب وجود برامج تناسب كل مجال من المجالات السابقة ، فمثلاً نحتاج لبرامج لإعداد الوجبات الغذائية المتكاملة ، و برامج تعليمية للمدارس و برامج حسابات و مخازن للتجار ، و برامج رسم هندسي للمهندسين ..الخ. كل هذه المهام – مهام تصميم البرامج – تقع على عاتق المبرمجين ، فهم المسئولون عن تصميم البرامج بالشكل الذي يناسب المستخدمين .

البرمجة تعني كتابة البرامج الحاسوبية بأسلوب علمي يضمن حلول حقيقة دقيقة للمسائل البرمجية و باستخدام إحدى لغات البرمجة ، وتمثل لغات البرمجة الأداة الأساسية المستخدمة في كتابة و تنسيق و ترجمة و تنفيذ البرامج .

تعريف النظام System definition :

النظام عبارة عن مجموعة من الوحدات – الأنظمة الفرعية أو النظيمات – التي تتكامل مع بعضها لإنجاز مهام محددة و كل وحدة أو نظام فرعي من وحدات النظام يوكل إليها جزء من المهام .

تعريف الحاسب Computer Definition :

الحاسب عبارة عن جهاز الكتروني يضم أجهزة كهربائية و ميكانيكية و معدات الكترونية يتلقى التعليمات من المستخدم و يقوم بإنجاز العمليات الحسابية و المنطقية ليقدم مخرجات يستفيد منها المستخدم ويقدم حلول و نتائج لدعم القرار.

مكونات نظام الحاسب Computer System Components :

يتكون نظام الحاسب الآلي من ثلاثة مكونات أساسية (تمثل أنظمة فرعية) هي:

المكونات المادية Hardware:

و تمثل الجزء الأساسي من نظام الحاسب الآلي و هي الأجزاء المادية الملموسة من النظام و تصنف إلى جزأين (أنظمة فرعية) :

- **وحدة النظام** System Unit،
- **الوحدات الطرفية** Peripheral Units
-

- **أولاً وحدة النظام** System Unit :

و هي الوحدة الأساسية في نظام المكونات المادية فهي تضم وحدة المعالجة المركزية المسئولة عن معالجة البيانات و وحدة الذاكرة الرئيسية المسئولة من تخزين البيانات الجاري تنفيذها في وحدة المعالجة المركزية و الذاكرة الثانوية التي تخزن فيها البيانات بشكل مستمر ، إذن مكونات وحدة النظام الأساسية هي:

1. وحدة المعالجة المركزية Central Processing unit (cpu)
2. الذاكرة الرئيسية Main Memory.
3. الذاكرة الثانوية Secondary Memory.

- **الوحدات الطرفية** Peripheral Units :

يعتبر كل جهاز غير وحدة النظام وحدة طرفية ، و تصنف الوحدات الطرفية إلى قسمين رئيسيين :

وحدات الإدخال Input Units :

و هي الوحدات المسئولة عن إدخال البيانات إلى الحاسب و تختلف وحدات الإدخال حسب نوع البيانات المدخلة فمنها ما هو خاص بإدخال البيانات النصية و منها ما هو خاص بإدخال البيانات الصوتية و الصورية ...الخ.

من وحدات الإدخال :

1. لوحة المفاتيح Key Board.
2. الفأرة الالكترونية Mouse.
3. الماسحة الضوئية Scanner.
4. الكاميرات الرقمية Camera.
5. المايكروفون Microphone.
6. الفاكس Fax.
7. المودم Modem

وحدات الإخراج Out Put Units.

و هي الوحدات المسئولة عن عرض النتائج و المخرجات و كذلك تختلف حسب نوع البيانات التي سيتم إخراجها من وحدات الإخراج :

- شاشة العرض Monitor.

- الطابعة Printer.
- السماعات الخارجية desktop Speaker.
- سماعات الأذن headphone Speaker.
- الفاكس Fax.
- المودم Modem

ملاحظة : بعض الأجهزة تعمل كوحدات إدخال و إخراج مثل الفاكس مودم.

المكونات البرمجية Software:

تضم منظومة البرمجيات الآتي:

- نظم التشغيل Operating System :
- لغات البرمجة Programming Languages.
- البرامج التطبيقية Applications .

أنظمة التشغيل عبارة عن مجموعة برامج تعمل كوسيط بين المستخدم و المكونات المادية ، تمكن المستخدم من استخدامها بسهولة و يسر، كما تمكنه من التحكم فيها و إدارتها ، من أمثلة نظم التشغيل :

- ويندوز Windows .
- لينكس Linux.
- يونكس Unix.
- دوس Dos.
- نوفل نتوير Novel Netware.
- سولارس Solaris .

أما لغات البرمجة فهي وسيلة التخاطب بين الإنسان و الحاسب ، وهي أداة بيد المبرمج يستخدمها لكتابة و تصميم و تنفيذ برامج لحل مشاكله البرمجية و هذه اللغات يمكن تصنيفها إلى:

1- لغة الآلة Machine Language و هي اللغة الوحيدة التي يفهما الحاسب ، و تتكون من أرقام من بين (0,1) و هي تختلف من حاسب لآخر .

2-لغة التجميع Assembly language : و هي لغة تستخدم اختصارات معبرة من اللغة الإنجليزية لتعبر بها عن العمليات الأساسية التي يقوم بها الحاسب من إضافة add و طرح sub و حفظ store و تتعامل مباشرة مع مجموعة مواقع في الذاكرة تسمى المسجلات Register.

3-لغات المستوى الأعلى High Level Language: و هي لغات تستخدم كلمات أقرب إلى لغة الإنسان مثل اللغة الإنجليزية ، هنالك الكثير من هذه اللغات مثل(بيسك basicو باسكال Pascal و فرتران Fortran سي و سي++ c/c++ ، و هنالك لغات أكثر تطوراً و هي لغات Visual مثل visual c++ و visual basic.. الخ.

- أما البرامج التطبيقية فهي برامج صممت بواسطة المبرمجين لحل مشاكل برمجية ، و تضم حزم البرامج الجاهزة ، التي تتولى شركات مثل مايكروسوفت إنتاجها مثل حزمة Office ، و برامج تطبيقات تصمم لحل مسائل برمجية بسيطة بواسطة لغات البرمجة .

المكونات البشرية Heartware :

هم الأشخاص الذين يتعاملون مع نظام المكونات و البرمجيات تختلف مهامهم فمنهم المبرمجون و منهم مهندسو النظم و محللي النظم و مدخلو بيانات و غيرهم.

البرنامج Program:

عبارة عن مجموعة من التعليمات مكتوبة بإحدى لغات البرمجة تعطى للحاسب الآلي ليقوم بعمل ما مثل حساب مجموع قيم رقمية.

المبرمج Programmer :

هو شخص ذو دراية و معرفة تامة بإحدى لغات البرمجة أو مجموعة منها و قادر على تحليل المشاكل البرمجية و تصميم حل مناسب لها باستخدام تلك اللغة أو إحدى تلك اللغات .

البرنامج المصدر Source Program:

هو البرنامج المكتوب بإحدى لغات البرمجة (لغات المستوى الأعلى مثل c/ c++ / /basic Pascal) و يمثل تعليمات برمجية لحل مسألة أو مشكلة ما.

البرنامج الهدف: Object Program:

هو البرنامج الناتج عن ترجمة البرنامج المصدر باستخدام مترجم لغة برمجة Compiler أو مفسر interpreter و يكون مكتوب بلغة الآلة و يمكن تنفيذه للحصول على النتائج .

أساليب البرمجة Programming Methods :

مرت عملية البرمجة بمراحل تطور مختلفة ابتداءاً من البرمجة بلغة الآلة – تتطلب البرمجة بلغة الآلة فهم المكونات المادية للحاسب فهم تام بالإضافة إلى فهم تعليمات لغة الآلة – و حتى البرمجة بلغات البرمجة كائنية التوجه OOP التي جعلت عملية البرمجة سهلة و بسيطة تتطلب فقط معرفة الكائنات و كيفية استخدامها بدلاً عن برمجتها فيما يلي سنوضح أساليب البرمجة المتبعة في كتابة و تصميم البرامج .

- **البرمجة الإجرائية** Procedural Programming .

في أسلوب البرمجة الإجرائية يكتب البرنامج كله كتلة واحدة في ملف واحد ، مما يجعل عملية البرمجة صعبة جداً لتداخل التعليمات و كثرتها فيصعب فهم البرنامج ويصعب معرفة الأخطاء اللغوية و المنطقية . من أمثلة اللغات التي تتبع أسلوب البرمجة الإجرائية إصدارات لغة البيسك الأولى (GW-Basic و BASICA).

- **البرمجة الهيكلية** Structural Programming .

أسلوب البرمجة الهيكلية غير نمط البرمجة الإجرائية بتقسيمه للبرنامج إلى مقاطع صغيرة و يعطي كل مقطع اسم معين و توكل إليه مهمة محددة و عند تنفيذ تلك المهمة يتم استدعاء ذلك المقطع ، هذه المقاطع تعرف بالبرامج الفرعية Sub Routines أو تعرف بالدوال و الإجراءات Functions & Procedures في بعض لغات البرمجة ، هذا التقسيم جعل من السهل فهم البرنامج و معرفة مكان الأخطاء اللغوية و المنطقية . و لكن إذا كبر البرنامج و تعقدت تعليماته و كثرت برامجه الفرعية (الدوال و الإجراءات) يكون من الصعب متابعة البرنامج و فهم تعليماته ، فكان أسلوب البرمجة بالأهداف الموجهة Object Oriented Programming (OOP).أمثلة للغات البرمجة الهيكلية لغة باسكال و لغة السي و لغات الفورتران و الكوبول .

- **البرمجة بالأهداف الموجهة** Object Oriented Programming :

في أسلوب البرمجة بالأهداف الموجهة (OOP) يتم تقسيم البرنامج إلى وحدات ذاتية الاحتواء تضم البيانات و مجموعة من البرامج الفرعية في كيان ، تسمى هذه الوحدات بالكائنات و كل كائن له صفات و له سلوك يميزه عن الكائنات الأخرى ، و تمثل البرمجة الكائنية عناصر البرنامج تمثيل حقيقي مطابق لتمثيل الكائنات العالم الحقيقي .

فوائد البرمجة بالأهداف الموجهة OOP benefits:

1- التجريد Abstraction (حماية و إخفاء البيانات) : إخفاء تفاصيل تصميم الكائن عن المستخدم أي استخدام الكائن دون الحاجة إلى معرفة تفاصيل تركيبه .
2- الكبسلة Encapsulation: وضع كل من البيانات و العمليات (الدوال) في مكان واحد يساعد المبرمج على التعامل مع الكائن بسهولة مثل نسخه وتعريفه .
3- إعادة الاستخدام Reuse (الوراثة inheritance) : يمكن للمبرمج إعادة استخدام كائن مرة أخري دون الحاجة إلى إعادة بناء الكائن من جديد مما يوفر الجهد و يزيد سرعة إنتاج البرامج ، و يمكن بناء كائن جديد يرث خصائص كائن آخر و يضيف إليها خصائصه.
4- تعدد الأشكال Polymorphism: من خلال تعدد الأشكال يمكن أن نجعل دالة ما تؤدي أكثر من وظيفة اعتماداً على الهدف الذي تتبع له.

المترجم Compiler:

من برامج النظم يقوم بترجمة البرنامج المصدر إلي برنامج بلغة الآلة قابل للتنفيذ ، وتتم ترجمة كل البرنامج دفعة واحدة و لا يتم تنفيذ البرنامج إلا بعد التأكد من خلوه من الأخطاء اللغوية .

المفسر Interpreter:

أيضاً من برامج النظم يقوم بترجمة البرنامج المصدر إلى برنامج بلغة الآلة قابل للتنفيذ ، و يختلف عن المترجم في أنه يقوم بترجمة التعليمات و تنفيذها تعليمة تلو الأخرى .

خطوات حل المسائل البرمجية (البرمجة):

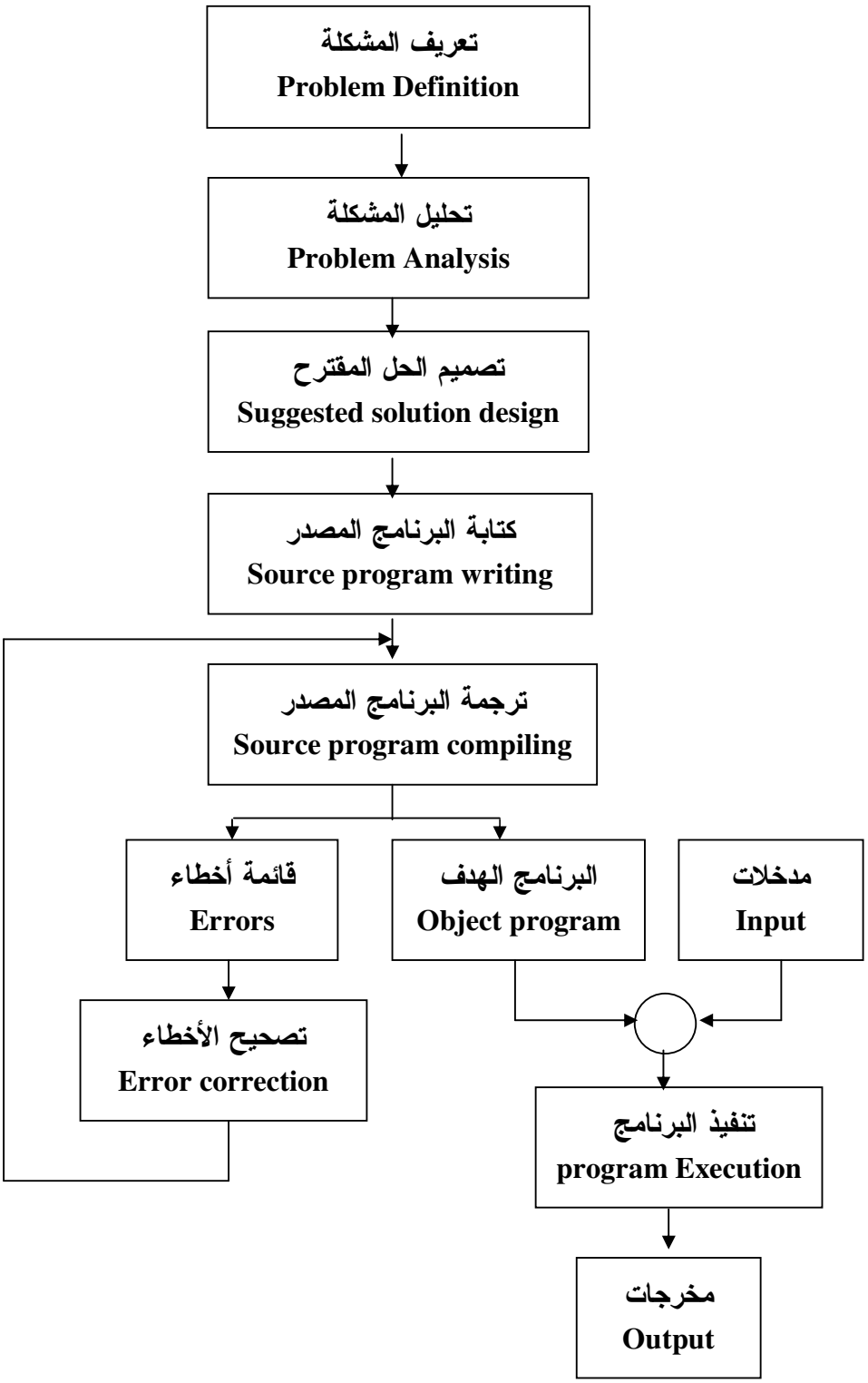
كما ذكرنا سابقا أن البرمجة تعني كتابة برامج باستخدام لغات البرمجة بصورة علمية تقود لحل المسائل البرمجية بصورة سليمة تضمن حلول أكيدة و موثوق بها ، وحتى نحصل على هذه الحلول الوثوق بها لابد من أن تمر عملية البرمجة بعدة مراحل نذكرها فيما يلي بالتفصيل :

1. تعريف المشكلة Problem Definition
2. تحليل المشكلة Problem Analysis.
3. تصميم الحل المقترح solution design.
4. برمجة الحل (كتابة البرنامج) solution Programming.
5. تنفيذ الحل – اختبار البرنامج Solution Implementation.
6. تشغيل البرنامج للحصول على الحلول و النتائج Program Execution.

يمكننا تقسيم الخطوات السالفة الذكر إلي مرحلتين ، الأولى تمثل دور الإنسان في حل المشكلة و الثانية تمثل دور الحاسب في حل المشكلة كالتالي:

- **المرحلة الأولى (دور الإنسان في حل المشكلة)** :
  - تعريف المشكلة .
  - تحليل المشكلة .
  - تصميم الحل المقترح .
- **المرحلة الثانية (دور الحاسب في حل المشكلة)** :
  - برمجة الحل المقترح .
  - تنفيذ الحل _اختبار البرنامج.
  - تشغيل البرنامج .

الشكل التالي (1−1) يبين خطوات حل المشكلة .



خطوات حل المسائل البرمجية

أولاً: تعريف المشكلة.

قبل البدء في حل المسائل البرمجية لابد من تعريف كل مسألة برمجية يراد إيجاد حل لها تعريفاً كاملاً ، و نقصد بتعريف المسألة فهمها فهماً تاماً و تحديد حدودها حتى لا يكون الحل ناقصاً أو غير كافياً أو أن يحيد الحل النهائي عن الحل المطلوب .

الكثير من المشاكل تبدو أكثر تعقيداً عن الحقيقة التي هي عليها و ذلك لعدم فهما فهماً عميقاً ، إذاً في هذه الخطوة يجب على المبرمج فهم المسألة و فهم كل جزيئاتها و كل ما يتعلق بها ، و تقسيمها إلى مشاكل فرعية بسيطة يسهل فهمها إن كانت معقدة.

ثانياً: تحليل المشكل :

و نعني بتحليل المشكلة تحليل المدخلات المطلوبة للمشكلة و معرفة كيفية معالجتها للوصول إلى الحلول المطلوبة و كذلك معرفة شكل المخرجات النهائية التي سيتم عرضها .

- تحليل المدخلات :

لابد من معرفة البيانات التي سيتم إدخالها للبرنامج كمعطيات لحل المشكلة و تحديد نوعها و حجمها مثلاً لإيجاد مجموع ثلاثة أعداد ، المعطيات لهذه المسألة ستكون ثلاثة أعداد يمكن تمثيلها ب X,Y,Z بحيث تمثل هذه المتغيرات أنواع رقمية بأقصى حجم يمكن أن تسمح به لغة البرمجة . إذا لم يتم الحصول على قيم هذه المتغيرات لن يكون هنالك معالجة أو مخرجات و نتائج .

- تحليل المعالجة :

للحصول على المخرجات لابد من معالجة البيانات التي تم إدخالها ، تحليل المعالجة يعني تحديد الطريقة التي سيتم عبرها الحصول على المخرجات ، مثلاً لمعالجة المسألة السابقة (إيجاد مجموع ثلاثة أعداد) فإننا سنستخدم المعادلة التالية لمعالجة المدخلات :

$$Sum=X+Y+Z$$

- تحليل المخرجات :

من خلال تحليل المخرجات سيتم تحديد كيفية عرض المخرجات بشكلها النهائي للمستخدم ، إذ لابد أن توافق المخرجات متطلبات المستخدم . في المسألة السابقة سيتم عرض قيمة المتغير SUM الذي تم حسابه سابقاً.

ثالثاً : تصميم الحل باستخدام الخوارزميات و خرائط التدفق :

هنالك العديد من الأساليب التي يمكن للمبرمج أن يستخدمها ليخطط حله المقترح ، من هذه الأساليب الخوارزميات ALGORITHS و مخططات التدفق FLOWCHARTS و الشفرة الزائفة PSEUDO CODE

تعريف الخوارزمية Algorithm Definition :

الخوارزمية عبارة عن خطوات مرتبة متسلسلة منطقياً تكتب بأي لغة بشرية لها بداية واحدة و نهاية واحدة تعبّر عن خطوات حل مسألة برمجية ، اسمها مشتق من اسم العالم المسلم محمد بن موسى الخوارزمي ، ويختلف حجمها باختلاف المسائل البرمجية ،و باختلاف الأشخاص الذين يقومون بكتابتها، يمكن وضع أكثر من خوارزمية لحل مسألة برمجية واحدة.

تمييز الخوارزميات بالصفات التالية :

1- لها بداية واحدة و نهاية واحدة.
2- مرتبة و متسلسلة منطقياً.
3- واضحة و بسيطة و غير غامضة .
4- توضح خطوات حل مسألة برمجية .
5- تكتب بأي لغة مفهومة .

أمثلة محلولة (1-1):

أكتب خوارزمية لحل المسائل البرمجية التالية :

1- إيجاد الوسط الحسابي لأربعة أعداد.
2- حساب مساحة الدائرة باستخدام $AREA= PIxR^2$
3- تحويل درجة الحرارة من فهرنهايت F إلى مئوي C بالعلاقة
$C=9/5*(F-32)$

الحلول :

أولاً الوسط الحسابي لـ4 أعداد.

1- البداية .

2- أدخل أربعة أعداد A,B,C,D.

3- احسب المجموع SUM=A+B+C+D.

4- اجعل AV=SUM/4.

5- اطبع الوسط الحسابي AV.

6- النهاية.

ثانياً مساحة الدائرة :

1- البداية .

2- أدخل نصف القطر R .

3- اجعل PI=3.14.

4- احسب المساحة AREA=PI*R*R .

5- اطبع المساحة AREA.

6- النهاية .

ثالثاً التحويل من فهرنهايت F إلى مئوي C:

1- البداية .

2- أدخل درجة الحرارة بالفهرنهايت F.

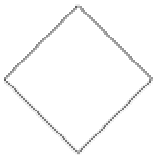
3- اجعل C=9/5*(F-32)

4- اطبع درجة الحرارة بالمئوي C.

5- النهاية .

مخططات التدفق Flow Chart :

تستخدم خرائط التدفق لبيان خطوات حل المشكلة و كيفية ارتباطها ببعضها ، باستخدام رموز اصطلاحية لتوضيح خطوات الحل و هذه الرموز مبينة بالشكل التالي:

معنى الرمز	**الشكل الاصطلاحي**
بداية أو نهاية START/STOP	
إدخال أو إخراج INPUT / OUTPUT	
معالجة PROCESSING	
قرار DECISION	
تكرار أو دوران LOOP	
نقطة توصيل و ربط CONNECTOR	
اتجاه سير البرنامج FLOW LINE	

شكل (1-2)

من أهم فوائم استخدام خرائط التدفق قبل كتابة البرنامج

1- تعطي صورة متكاملة للخطوات المطلوبة لحل المشكلة .

2- تمكن المبرمج من الاحاطة التامة بكل أجزاء المسألة .

3- تساعد المبرمج على تشخيص الأخطاء ، وخاصة الأخطاء المنطقية.

4- تيسر للمبرمج أمر إدخال أي تعديلات في أي جزء من المسألة.

أنواع خرائط التدفق :

هنالك نوعان رئيسيان من خرائط العمليات :

- **خرائط سير النظم** SYSTEM FLOWCHARTS:

يستخدم هذا النوع من الخرائط عند تصميم الأجهزة الهندسية في المصانع و غيرها و التي تستخدم أنظمة ذاتية التحكم .

- **خرائط سير البرامج** PROGRAMS FLOWCHARTS:

و يستعمل هذا النوع من الخرائط لبيان الخطوات الرئيسية التي توضع لحل مسألة ما و ذلك بشكل رسوم اصطلاحية تبين العلاقات المنطقية بين سائر خطوات الحل .و يمكن تصنيف خرائط سير البرامج إلى ثلاثة أنواع رئيسية :

1. خرائط التتابع البسيطة SIMPLE SEQUENTIAL FLOWCHART
2. الخرائط ذات الفروع BRANCHED FLOWCHARTS.
3. خرائط الدوران LOOP FLOWCHART.

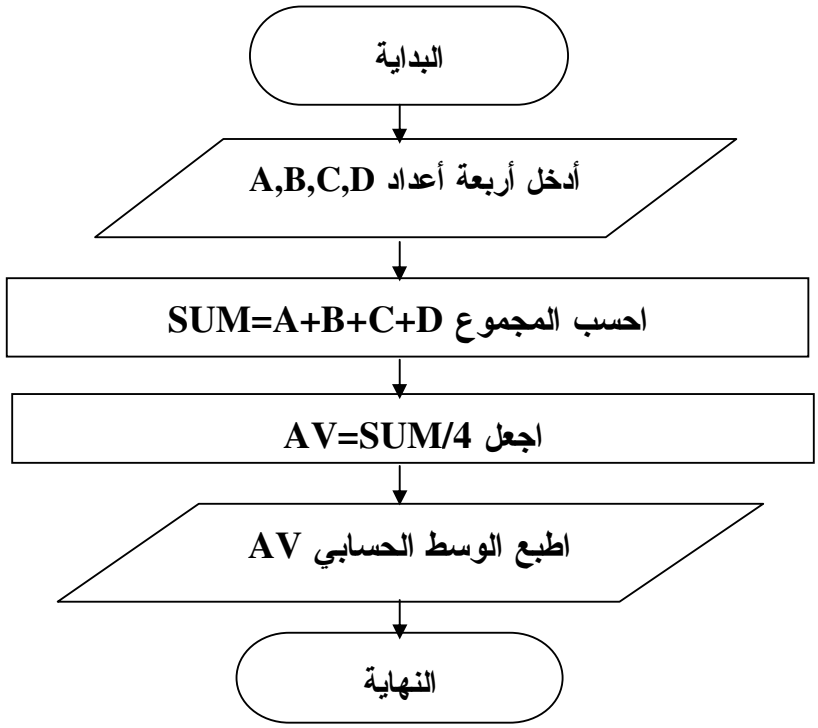
- **أولاً : خرائط التتابع البسيطة :**

في خرائط التتابع البسيطة تكون المسألة بسيطة غير معقدة الخطوات ، و تكون خطوات حلها متسلسلة لا يوجد بها تكرار لعملية ما أو اختيار و تفرع ، مثال لهذه المسائل البرمجية المسائل الثلاثة المذكورة آنفاً ،أدناه أمثلة المخططات التدفقية ذات التتابع البسيط .

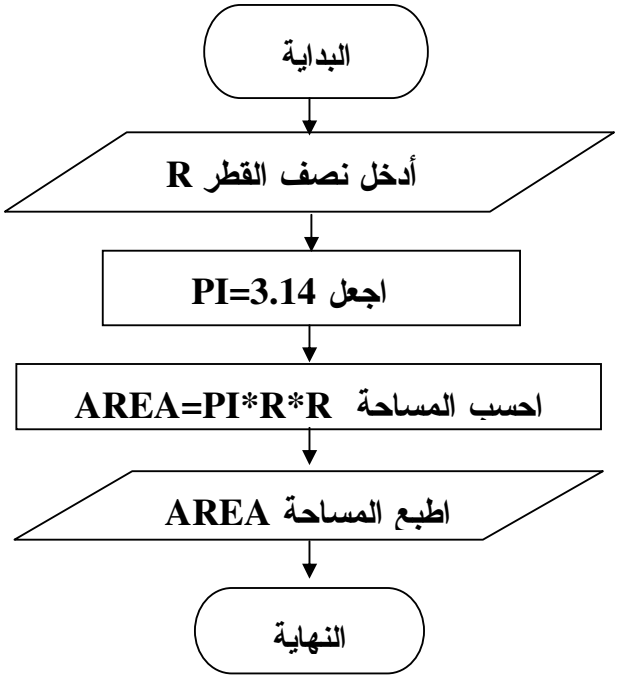
أمثلة محلولة(1-2) : أرسم مخطط التدفق للمسائل في (1−1)

الحلول:

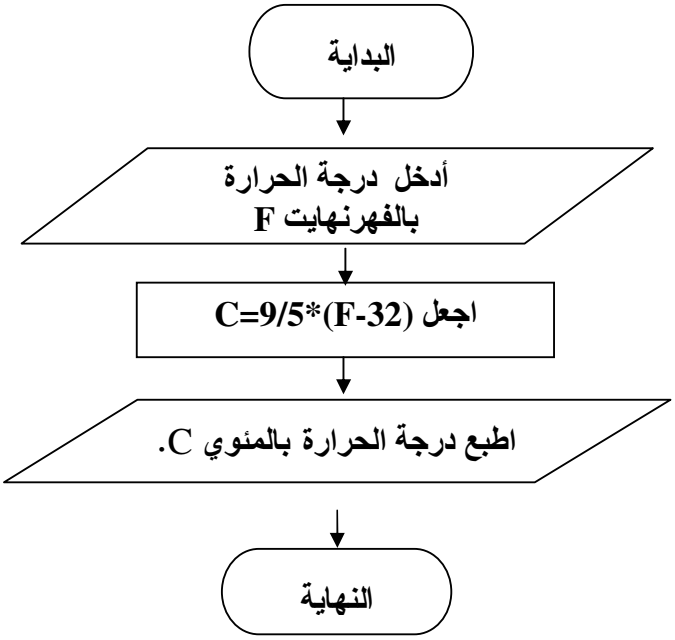
أولاً : الوسط الحسابي لأربعة أعداد :



ثانياً : حساب مساحة الدائرة :



ثالثاً التحويل من فهرنهايت F إلى مئوي C



- **ثانياً : الخرائط ذات الفروع:**

أما المخططات ذات الفروع ، فمثل خرائط لمسائل برمجية معقدة قليلاً، و تحتوي على عمليات تتطلب الاختيار و التفرع المثال التالي يبين شكلاً من هذه الأشكال:

مثال: أرسم مخططاً تدفقياً لمقارنة رقمين و طباعة الرقم الأكبر:

أولاً الخوارزمية :

1- البداية .

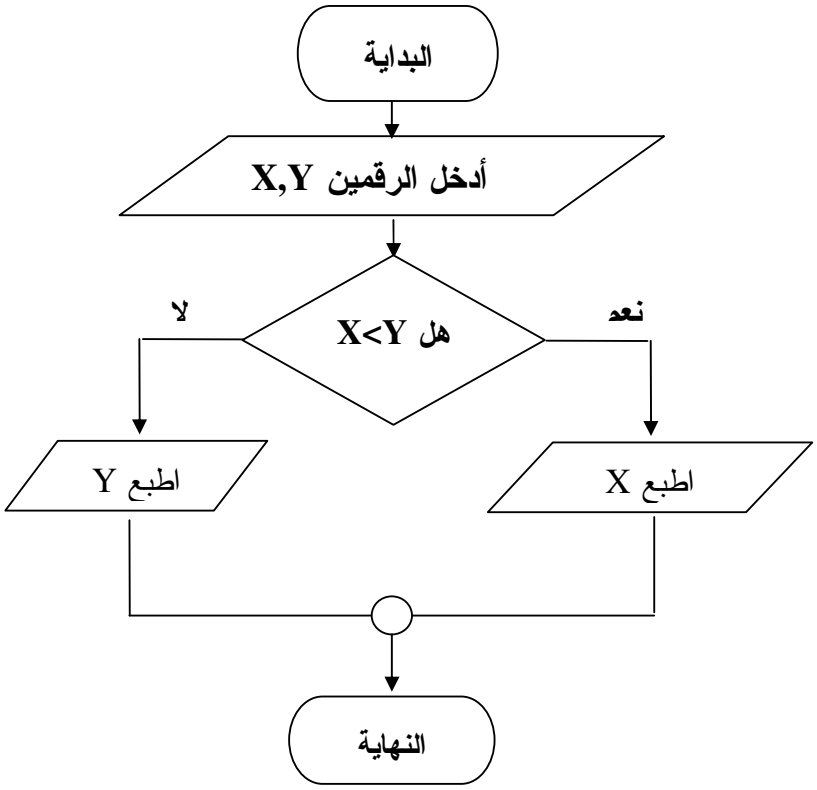
2- أخل الرقين للمقارنة.

3- هل X>Y .

4- إذا كان الناتج نعم اطبع X ثم اذهب إلى الخطوة 6.

5- اطبع Y.

6- النهاية.



ملاحظة: دائما قبل رسم المخطط الانسيابي لابد من كتابة الخوارزمية لتسهل عليك رسم المخطط.

ثالثاً: خرائط الدوران

بعض المسائل البرمجية تتطلب تكرار عملية معينة عدة مرات ، مثلا ، في مسألة برمجية ما ، نريد أن نكرر تعليمة 100 مرة ، ليس من المنطقي أن نقوم بكتابة 100 خطوة أو رسم 100 خطوة في خريطة التدفق ، و لكن يمكن كتابة خطوة واحدة ثم تكرار هذه الخطوة مائة مرة . المثال التالي يبين هذا النوع من المخططات.

أمثلة محلولة (1-3):

أكتب خوارزمية ثم ارسم خريطة تدفق للمسألة البرمجية التالية:

طباعة الأرقام من 1- 100 على الشاشة بصورة متسلسلة .

الحل:

أولاً الخوارزمية :

1- البداية .

2- اجعل i=1 .

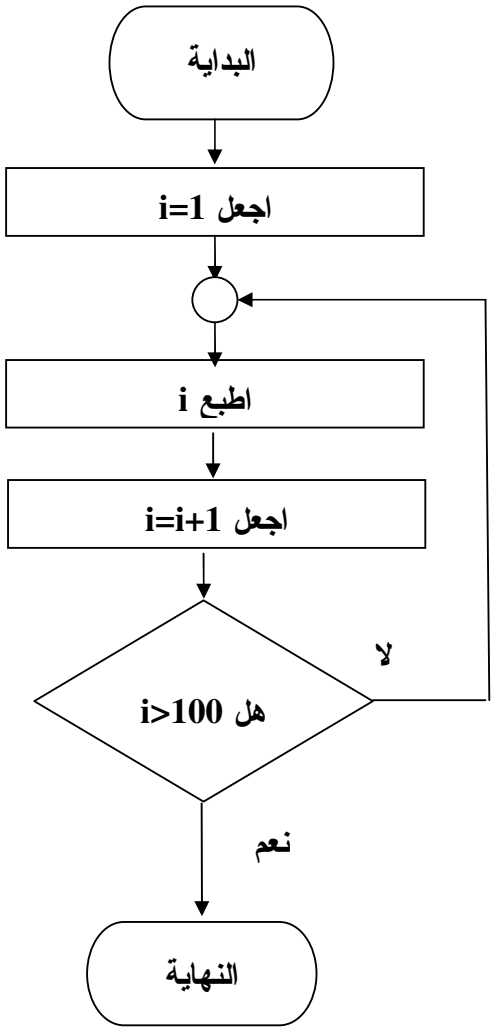
3- اطبع i.

4- اجعل i=i+1.

5- هل I>100 إذا كان لا اذهب إلى الخطوة 3.

6- النهاية.

ثانيا: المخطط الانسيابي (خريطة التدفق):



رابعاً: كتابة البرنامج :

بعد تعريف المشكلة تعريفاً كاملاً و تحديد و تحليل المدخلات و المخرجات ، ثم كتابة الخوارزميات و بناء خرائط التدفق ، يقوم المبرمج بكتابة شفرة البرنامج باستخدام إحدى لغات البرمجة التي يجيدها ، ثم ينقل هذا البرنامج إلى الحاسب ليمثل البرنامج المصدر Source Program ليقوم بترجمته إلى لغة الآلة – البرنامج الهدف Object Program – مستخدماً مترجم اللغة ،خلال عملية الترجمة قد تواجه المبرمج بعض الأخطاء اللغوية – كأخطاء في كتابة تعليمات برمجية – أو أخطاء منطقية – كأخطاء في تسلسل تعليمات البرنامج ، مما يطره إلى تصحيح هذه الأخطاء ، بعدها يصبح البرنامج جاهزاً لتجربته و التحقق من قدرته على إعطاء حلول معقولة و صحيحة منطقياً .

خامساً: تنفيذ البرنامج (اختبار الحل) Solution Implementation:

هذه الخطوة من أهم الخطوات ، فبعد التأكد من خلو البرنامج من الأخطاء المنطقية و اللغوية سيتم اختبار البرنامج بمدخلات بسيطة معلومة القيمة للتأكد من أن البرنامج يعمل بصورة سليمة. و كذلك للتأكد من أهن يعطي الحلول المطلوبة .

سادساً : تشغيل البرنامج بمعطيات حقيقة:

الخطوة الأخير في عملية البرمجة و هي تنفيذ البرنامج باستخدام القيم و المدخلات الحقيقة التي تمثل مدخلات المسالة البرمجة التي من اجلها كتب البرنامج ، يتبع لهذه الخطوة أيضا إضافة التعليقات و العبارات التي من شأنها إزالة اللبس و الغموض عن بعض الجمل البرمجية ، و لمساعدة من يستخدم البرنامج من بعدك في عمليات التعديل و الترقية و الصيانة ، تمسى هذه العملية بالتوثيق .

نهاية الجزء الأول بحمد الله