جامعة الملك سعود

مركز التدريب وخدمة المجتمع

دبلوم علوم الحاسب الآلي التطبيقي

أساسيات البرمجة

تطور لغات البرمجة

1-1 مقدمة عن لغات البرمجة

نعلم أن دراسة علوم الحاسب تنقسم الى قسمين هما

- العتاد Hardware
- البرمجيات Software

ولتسهيل الدراسة يتم تجزئة كل قسم على حده.فتم تقسيم العتاد الى وحدات الإدخال، وحدات الإخراج، وحدة النظام (المكونة من وحدة المعالجة المركزية ووحدة الذاكرة). وتم تقسيم البرمجيات الى نظم التشغيل،لغات البرمجة، البرامج التطبيقية.

لغات البرمجة :

تنقسم بصفة عامة الى مستويين أساسيين هما:

- لغات المستوى المنخفض
- لغات المستوى العالي

وبالطبع هناك فارق كبير بين هذين المستويين في الإمكانيات، وسهولة التعامل مع الحاسب، بالإضافة الى سهولة تعلم اللغة وفهمها. وبما أن لغات المستوى العالي تستخدم كلمات إنجليزية معينة ورموز رياضية مألوفة، فهي أسهل في تعلمها وفهمها.

1-1-1 لغات المستوى المنخفض

تنقسم الى قسمين هما :

- لغة الآلة
- لغة التجميع

- لغة الآلة

هي أول اللغات ظهور، وهي اللغة الوحيدة التي يفهمها الحاسب مباشرة دون وسيط، وتتكون من رمزين هما الصفر والواحد. هذان الرمزان يعبران عن الأوامر المختلفة والبيانات التي يتكون منها البرنامج. إلا أن هذه اللغة صعبة التعلم وخاصة أن لكل حاسب لغة آلة خاصة به، وتتطلب معرفة واسعة في تصميم الحاسب بالغضافة الى صعوبة إكتشاف الأخطاء. مما أدى الى تطوير هذه اللغة الى لغة التجميع .

- لغة التجميع :

تعتمد على الرموز المختزلة ، أي اختصارات الكلمات ذات مدلول لغوي محدد مثل : ADD تدل على الجمع، و MOV تدل على النقل ، وهكذا ...، مما جعل تعلم هذه

اللغة اسهل نسبياً من لغة الآلة، بالإضافة الى سهولة الكشف عن الأخطاء وتصحيحها. ولكن البرنامج في هذه اللغة يتم تجميعه وتحويله الى لغة الآلة عن طريق ما يسمى بالمجمع . مما يؤكد أن الحاسب لا يتعامل مباشرة إلا مع لغة الآلة.
ومع ذلك تبقى هذه اللغة صعبة التعلم ، ولها عيوب من أبرزها ارتباطها بالآلة ، فكل آلة لها لغة تجميع خاصة بها ، ويقصد بالآلة هنا تحديداً المعالج .
بناء على ما سبق نقول إن كتابة البرامج بلغات المستوى المنخفض تعتمد على معرفة واسعة بالتصميم الداخلى للحاسب (المعالجات، المقاطعات، مسارات البيانات ، عناوين الذاكرة ....) مما جعل العلماء يفكرون في لغات تعزل المبرمج نسبياً عن التصميم الداخلى للحاسب.

2-1-1 لغات المستوى العالي :

تعتمد على كلمات إنجليزية واضحة المدلول مثل : write, read, input , print ، وتم عزل المبرمج عن مشقة الخوض في متاهات التصميم الداخلي للحاسب، مما سهل تعلم هذه اللغات والإقبال عليها لحل المشكلات والتطبيقات العلمية والتجارية وغيرها . إلا أن تنفيذ البرنامج بهذه اللغات يحتاج الى كشف الأخطاء وتتبع التعليمات خطوة خطوة وذلك عن طريق ما يسمى بالمفسرInterpreter ثم ترجمته وتحويله الى لغة الآلة عن طريق ما يسمى بالمترجم Compiler.
والآن يمكن إيجاز مميزات لغات المستوى العالي فيما يلي :

- عدم الإرتباط بمعالجات معينة مثل لغات التجميع، وذلك لأن هذه اللغات مصممة اساساً لحل نوعية محددة من المشاكل وليست لنوعية محددة من المعالجات .
- سهولة تعلمها وسهولة كتابة البرامج بها، وذلك لإستخدامها كلمات وتعبيرات مشابهة لما يستخدمه الإنسان .
- سهولة اكتشاف الأخطاء وتصحيحها .
- توفير الجهد والوقت الذي كان يقوم به المبرمجون عند كتابة البرامج بلغة الآلة أو لغة التجميع.

أما اللغات التي ظهرت في هذا المستوى فهي كثيرة جداً ، من أبرزها وأشهرها : الفورتران ، الكوبول ، البيسك ، الباسكال ، السى ، ... الخ

ويمكن أن نطلق على هذه اللغات بأنها لغات خطية بسبب كونها تعتمد على التعليمات والأوامر المتسلسلة و المرتبة والتي تتوافق مع الخوارزمية . وقد تفاضلت هذه اللغات فيما بينها من حيث القوة والسهولة فكانت لغة البيسك Basic هي اللغة الأكثر شهرة وشعبية، وعامة الاستخدام لجميع المبتدئين في البرمجة ، وقد اشتقت حروفها من الحروف الأولى للعبارة الآتية :

Beginners All-Purpose Symbolic Instruction Code

وتعنى لغة التعليمات الرمزية المتعددة الأغراض للمبتدئين. ظهرت هذه اللغة في الستينات في كلية جامعية في الولايات المتحدة الأمريكية، ثم طورت من قبل معهد المقاييس الأمريكية ANSI عام 1968م، ومن أهم الإصدارات كانت QBasic .

بقيت هذه اللغات البرمجية بكافة أنواعها ضعيقة من حيث واجهات البرنامج التي تنشئها، والواجهات المقبولة تتطلب كتابة آلاف الأسطر اثناء تصميم البرنامج، مما دفع الشركات لتطوير هذه اللغات الى لغات مرئية، وخصوصاً يعد ظهور نظام النوافذ Windows الذي يدعم البيئة الرسومية، في هذه الآونة ظهرت اللغات المرئية مثل فيجوال بيسك ، الدلفى (فجوال باسكال)، فبجوال سي ++ ...الخ.

وقد تبنت شركة مايكروسوفت لغة البرمجة QBasic لتكون نواة للغة بيسك المرئية(فيجوال بيسك) Visual Basic ، وقد ظهر اول اصدار لها عام 1991م ومايزال التحديث جارياً على هذه اللغة حتى الآن.

وأصبحت لغة Visual Basic في اصداراتها الحديثة مصممة للكائنات .

تبين لنا مما سبق أن لغات البرمجة عالية المستوى تنقسم الى قسمين هما :

- لغات خطية : تعتمد على كتابة الأوامر والتعليمات على شكل خطوات مرتبة ومنتهية، وهذه الخطوات تمثل ترجمة للخوارزمية، مثل لغة البيسك، الفورتران ، باسكال، وغيرها.
- لغات مرئية: تعتمد على الكائنات أو الأدوات والأحداث، حيث يتم الترابط بين الكائنات بعمليات برمجية، ثم يتم تنفيذ المشروع عن طريق الأحداث ، ولذلك يسمون لغة بيسك المرئية باللغة الموجهة بالأحداث أو اللغة المسيرة بالأحداث، أي اللغة التي تنفذ تعليماتها واجراءاتها عند اختيار الاحداث، والحدث هو كل تأثيريتم على الفأرة أو لوحة المفاتيح، مثل Double click او Click أو غيرها من الأحداث .

تطور أساليب البرمجة :

لقد مرت اساليب البرمجة بثلاث مراحل:

- البرمجة العشوائية
- البرمجة الهيكلية
- البرمجة غرضية التوجه

1. البرمجة العشوائية :

يركز هذا الاسلوب من البرمجة على حل المسألة برمجياً وتحقيق الهدف دون النظر الى عملية تنظيم البرنامج، وفي هذه الحالة يعاني البرنامج من صعوبة في التطوير وفي اكتشاف الاخطاء، وربما حصل تكرار في بعض المقاطع البرمجية، كما ان استخدامالامر Go to بكثرة يعيق فهم البرنامج وصعوبة تتبع خطوات التنفيذ، وفي هذه الحالة ينظر للبرنامج كأنه كتلة واحدة . وهذاالاسلوب لا تظهر مشاكله الا اذا كان البرنامج كبيراً وضخماً، اما في حالة البرامج التدريبية البسيطة ربمايكون مناسباً،لانه لا يحتاج إلى تجزئة.

2. البرمجة الهيكلية :

يعتمد هذا الاسلوب من البرمجة على تجزئة البرنامج الى عدة برامج فرعية، حيث يتم الربط بين هذه البرامج الفرعية لتشكل البرنامج العام . والمقصود بالبرامج الفرعية function او procedure . يحتاج هذاالاسلوب الى تخطيط جيد، وتظهر فاعليته في حالة المسائل المتوسطة الحجم والتي تطرح على الطلاب في المرحلة الجامعية، كما ان هذا الاسلوب يسهل من اكتشاف الأخطاء، واجراء عمليات التطوير، وعدم تكرار المقاطع.

3. البرمجة غرضية التوجه :

تدعى ايضا بالبرمجة الشيئية او الكائنية المنحنى وهي البرمجة التي تحاكي الواقع . ويعتمد هذا الاسلوب على بناء الكائنات التي تضم البيانات والاجراءات وجملة ترابط الكائنات تشكل المشروع .

ما هي البرمجة غرضية التوجه ؟

ظهرت البرمجة غرضية التوجه في بداية السبعينات من هذا القرن، حيث بدت الحاجة ماسة لها، بعد ان واجهت البرمجة الهيكلية (الاجرائية) عدة مشاكل منها : البيانات غير المحمية، عدم القدرة على محاكاة الواقع، صعوبة تقسيم البرنامج الى اجراءات، عدم القدرة على اعادة الاستعمال، وغيرها من الاسباب.

قد يتصور البعض ان البرمجة غرضية التوجه انما هي لغة برمجة جديدة، ولكن الامر ليس كذلك، انما هي اسلوب تنظيمي في البرمجة بكافة توجهاتها وعلى اختلاف لغاتها، وان استخدام اسلوب البرمجة غرضية التوجه لا يعني نسف كل ما تعلمناه كمبرمجين، لا بل هو الالمام بالامكانات المتاحة من خلال هذا الاسلوب.

ان مصممي البرمجة غرضية التوجه اعتبروا انالاجسام من حولنا ما هي الا كائنات تتفاعل مع بعضها وفق علاقات تتفق مع طبيعتها، بغض النظر عن كون هذه الكائنات حية ام جامدة، والهدف من ذلك هو مساعدة المبرمجين على كتابة برامج قابلة للتطويروفي مدة زمنية أقل.

لقد مر علينا ذكر الكائن (Object) فما هو الكائن وفق ما تراه البرمجة غرضية التوجه ؟

الكائن (Object) :

هو نمط جديد (متغير مركب يشبه نسبياً السجل في تعريفه البرمجي) ينتج من دمج البيانات والإجراءات التي تعمل على تلك البيانات في كينونة واحدة .

الآن عندما تحاول حل مشكلة في البرمجة غرضية التوجه، لن تتساءل عن كيفية تقسيم المشكلة الى إجراءات او دوال، بل عن كيفية تقسيمها الى كائنات .أن التفكيربالكائنات بدلا عن الاجراءات له تأثير مفاجىء عن مدى سهولة تصميم البرنامج بهذا الاسلوب، وذلك بسبب المطابقة القوية بين الكائنات في المفهوم البرمجي والكائنات في الحياة الفعلية.

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 :

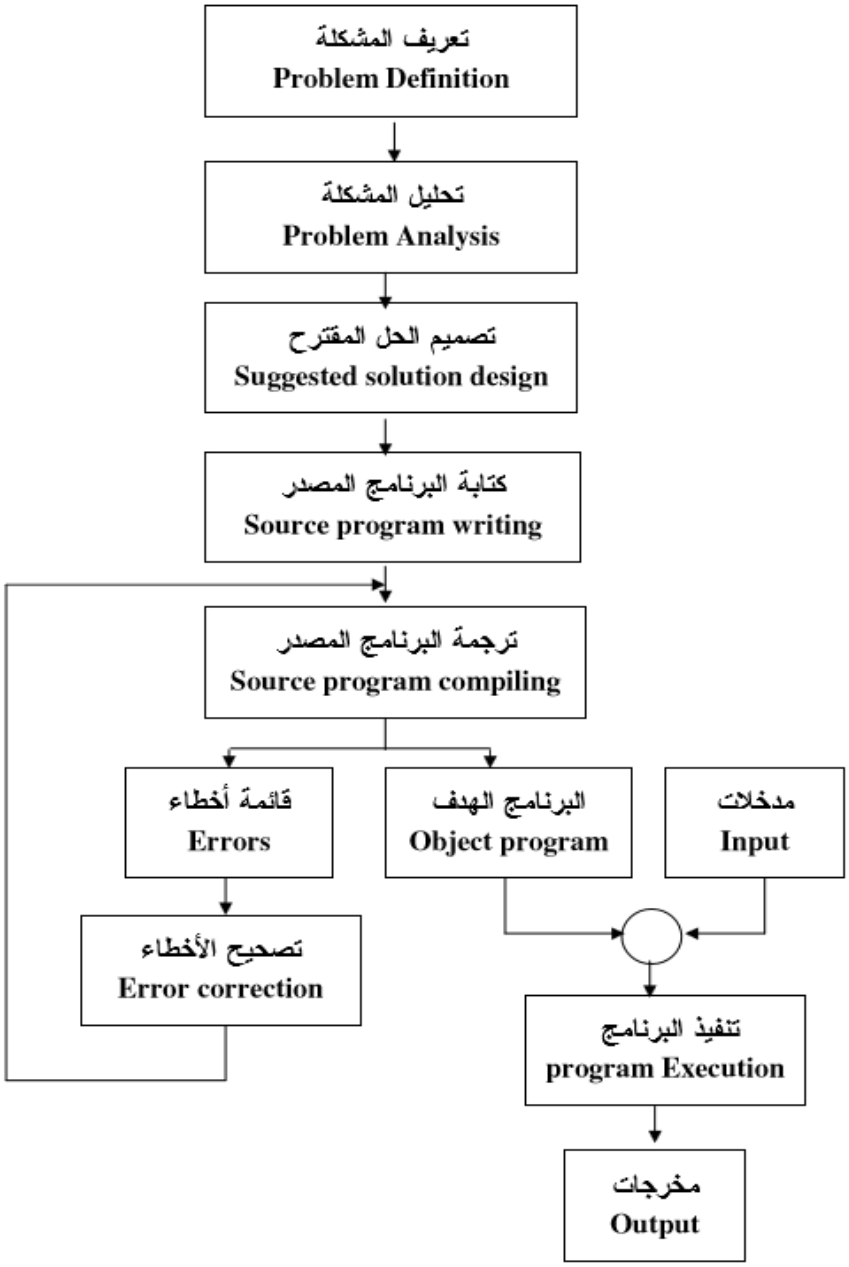
1. تعريف المشكلة Problem Definition
2. تحليل المشكلة Problem Analysis.
3. تصميم الحل المقترح solution design.
4. برمجة الحل (كتابة البرنامج) solution Programming.
5. تنفيذ الحل – اختبار البرنامج Solution Implementation.
6. تشغيل البرنامج للحصول على الحلول و النتائج Program Execution.

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 .



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.

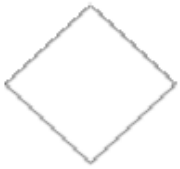
3- اجعل C=9/5*(F-32)

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 فوائد 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.

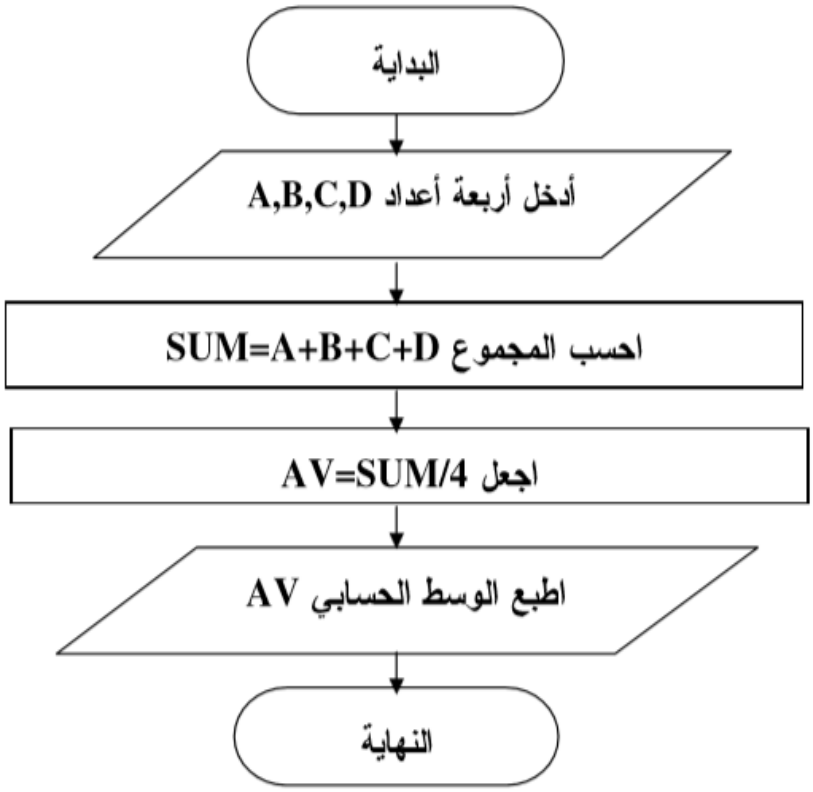
- **أولاً : خرائط التتابع البسيطة :**

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 .

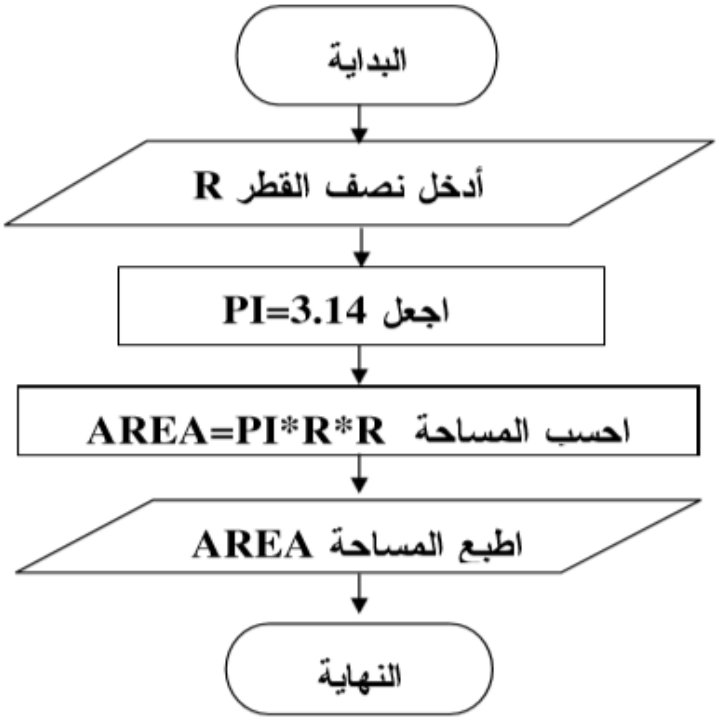
أمثلة محلولة(1-2) : أرسم مخطط التدفق للمسائل في (1-1)

الحلول:

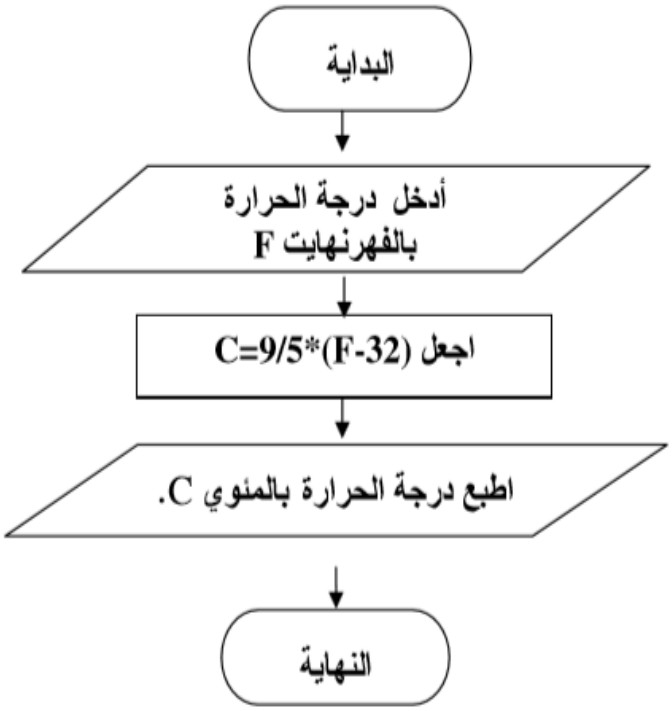
أولاً : الوسط الحسابي لأربعة أعداد :



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 .

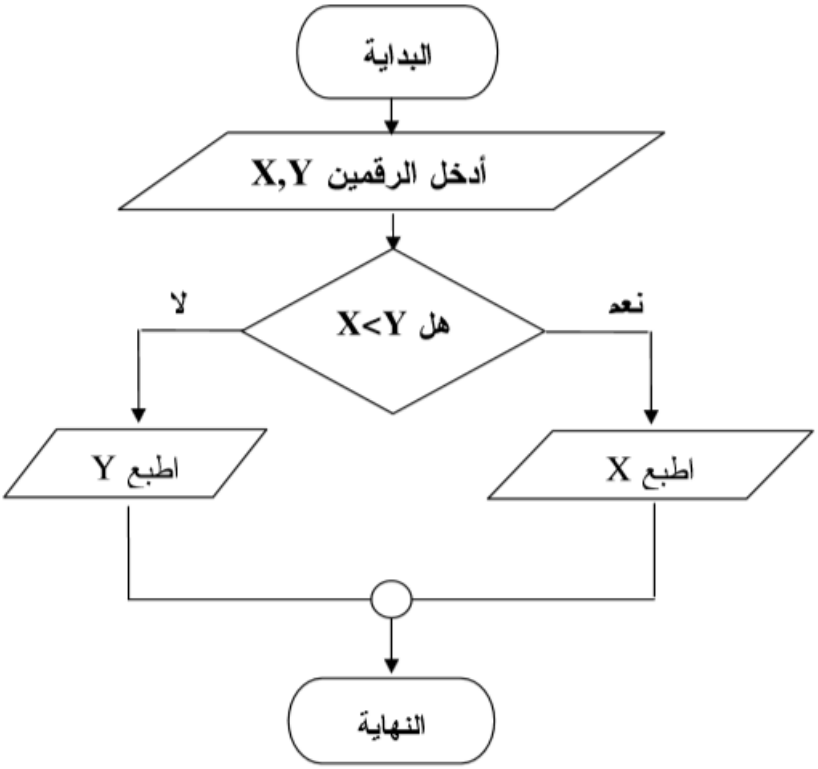
2- أخل الرقين للمقارنة.

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 .

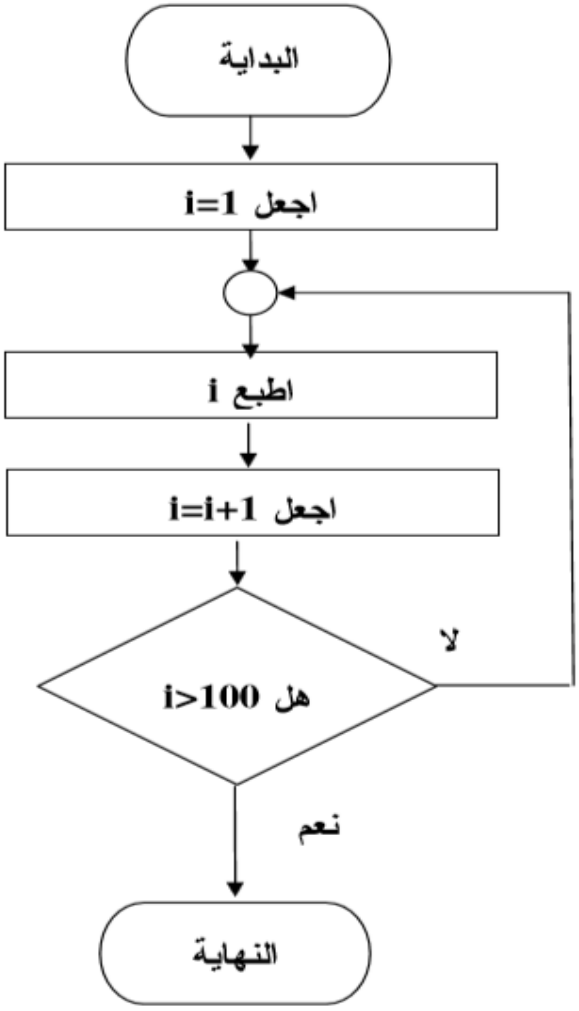
3- اطبع i.

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 .

فيجوال بيسك (Visual BASIC)

هي بيئة تطوير ولغة برمجة من مايكروسوفت تستند إلى لغة البيسك الشهيرة. وهي تصنف ضمن لغات البرمجة بالكائنات. منذ أن بدأت مايكروسوفت في اصدار الفيجوال بيسك وهي تلاقي نجاحا باهرا وشعبية لا بأس بها بين المبرمجين نظرا لسهولتها الشديدة في مقابل التعقيد الشديد الذي يواجهه أي مبرمج يسعى لبرمجة ويندوز باستخدام السي أو السي++ .عموما تناسب الفيجوال بيسك تطبيقات قواعد بيانات والتطبيقات المخصصة للشركات الصغيرة وبرامج الحسابات وهي مريحة وسهلة وتؤدي الغرض بالإضافة إلى أنها تسمح للمبرمج بالتركيز على حل المشكلة فغالباً ما لا يواجه صعوبات فنية أثناء كتابة برنامج بالفيجوال بيسك

لغة البرمجة فيجوال بيسك هي لغة ذات تصميم مرئي واجهة رسومية بعكس بعض اللغات مثل (الاسمبلي) ذات الشاشة السوداء. حيث تحتوي هذه اللغة على العديد من الاوامر بداخلها ولغة سهلة التطبيق تم تطوير هذه النسخة من البرنامج عن النسخة القديمة basic والتي تعمل تحت بيئة dos إلى هذه النسخة التي تعمل تحت بيئة ويندوز.

التاريخ:

أنتجت شركة مايكروسوفت أول إصدار من لغة البيسك عام1975 م، وسمي Basic والاسم يعتبر اختصاراً لكلمة لغة البرمجة العامة التسلسلية للمبتدئين (Beginner's All-Purpose Symbolic Instruction Code)، وتوالت الإصدارات ، ومع انتشار بيئة ويندوز ظهرت فيجوال بيسك التي احتوت على كثير من أوامر QBASIC وأضيفت العديد من الوظائف التي جعلت من البرمجة بفيجوال بيسك يسرة وسهلة. هناك الآلاف من المواقع التعليمية للفيجوال بيسك ومنها Visual Basic Tutorials

في عام 2000 قامت مايكروسوفت بإنتاج النسخة المطورة VISUAL BASIC.NET والتي تعتمد على البرمجة الشيئية.

مميزات الفيجوال بيسك

- لغة سهلة وسريعة لإنشاء تطبيقات ويندوز
- تدعم البرمجة الشيئية إلا أن ذلك ليس بشكل كامل.
- تعتبر لغة الفيجوال بيسك لغة كائنية المنحنى
- سهلة التعلم والفهم
- سهولة اكتشاف الأخطاء فيها
- اعتماده على HTML وذلك مما جعله سهل الاستعمال والفهم.
- عند كتابة اوامر صحيحة يقوم بإعطائك أمثلة ليؤكد لك على صحة كتابة الكود
- تمكنك من تخطي بعض الاخطاء عند كتابة كود محدد

إصدارات فيجوال بيسك

فيجوال بيسك 1 (1991م)

الإصدار الأول من Visual Basic كان محدود للغاية. ولم يكن موجه لتطوير التطبيقات الحقيقة لبيئة .Windows مع انه كان سهل الاستخدام ذو واجهة رسومية ولغة برمجة مرئية إلا انه كان يعتبر كلعبة مسلية للمبرمجين.

فيجوال بيسك 2 (1992م)

الإصدار الثاني من Visual Basic لم يظهر أي جديد باستثناء إضافة القليل من الخصائص ودعم أفضل لبيئة التطوير المتكاملة .IDE

فيجوال بيسك 3 (1993م)

يعتبر الإصدار الثالث هو بداية طريق النجومية أو الشهرة لـ !Visual Basic حيث قدم دعم لقواعد البيانات وأصبح ذو نهاية مفتوحة بفضل الإضافات التي كنت تستطيع دمجها حيث ظهرت الكثير من التحسينات في بيئة التطوير المتكاملة IDE وهاجر مئات إن لم يكن آلاف المبرمجين إلى .Visual Basic وبدأت تلك اللغة كمنافس ضعيف لتطوير البرامج الحقيقية أو التجارية لأنه كان ما زال ينقصها المزيد.

فيجوال بيسك 4 (1995م)

كان الهدف الأساسي من الإصدار الرابع هو مرحلة انتقالية إلى Windows 95 أو إن صح التعبير، القابلية لتطوير تطبيقات من نوع bit. 32 وكان أول إصدار من إصدارات Visual Basicتولد شيفرة للعمل تحت معالجات من نوع Bit. 32

فيجوال بيسك 5 (1997م)

الإصدار الخامس كان بمثابة الإعلان الرسمي في أن لغة Visual Basic هي لغة برمجة لتطوير التطبيقات الحقيقية والتجارية..

فيجوال بيسك 6(1998م)

الإصدار السادس لا يختلف عن الإصدار الخامس كثيرا لكن هناك العديد من التحسينات وعلاج للأخطاء التي كانت موجودة في الإصدار الخامس. من أهم الإضافات في الإصدار السادس هي الزيادة في أدوات قواعد البيانات والمبنية على .ADO كذلك تحسن واضح في أدوات التحكم.

فيجوال بيسك.نت

تم إنتاج هذا الإصدار مع تغيير جذري عما سبقه من الإصدارات

ولقد صدر في ظل هذه التقنية سبعة إصدارات إلى الآن

- **فيجوال بيسك.نت 2002**
- **فيجوال بيسك.نت 2003**
- **فيجوال بيسك.نت 2005**
- **فيجوال بيسك.نت 2008**
- **فيجوال بيسك.نت 2010**
- **فيجوال بيسك.نت 2012**
- **فيجوال بيسك.نت 2013**
- **فيجوال بيسك.نت 2015**
- **فيجوال بيسك.نت 2017**

أنواع البيانات

لو نظرنا إلى مراحل العمل فى الحاسب سنجد أنها عبارة عن ٣ خطوات أساسية وهى (إدخال - معالجة - إخراج) ، تُعرف المرحلة الأولى بمرحلة إدخال البيانات والمرحلة الثانية هى معالجة البيانات وإجراء العمليات الحسابية والمنطقية عليها ثم مرحلة إخراج النتائج أو المعلومات ، والبيانات هى عبارة عن قيم أوليه تُجرى عليها بعض العمليات لتصبح معلومات أى قيم لها معنى ومدلول يمكن الاستفادة منها ، إذن فجميع مراحل العمل داخل الحاسب تحتوى على بيانات والبيانات لها صور متعددة فيمكن أن تكون أرقام أو حروف أو علامات أو صور أو صوت أو فيديو ، هذه الصور جميعا يتعامل معها الحاسب على هيئة أرقام بالنظام الثنائى (لغة الآلة) وتخزن بالذاكرة أثناء فترة العمل عليها ، بناء على ما ذكرناه فإن أى برنامج يتعامل مع بيانات بحيث يستقبل بيانات من المستخدم ويرسلها للحاسب ثم يستقبل النتائج من الحاسب ويعرضها للمستخدم.

أنواع البيانات Data Types

فى لغات البرمجة إذا أردت أن تتعامل مع بيانات معينة يجب عليك أن تحجز لها مكانا فى ذاكرة الحاسب حتى يستطيع الحاسب التعامل معها ، هذه البيانات يمكن أن تكون أعداد والأعداد تختلف من حيث النوع فمنها الصحيح ومنها الحقيقى (الذى يحتوى على فاصلة عشرية) ويمكن أن يكون العدد قيمته صغيرة أو كبيرة ، والبيانات أيضا يمكن أن تكون نصوص والنص يحتوى على حروف أو أرقام أو علامات أو خليط بينهما ، من هذا الاختلاف تم تقسيم البيانات إلى أنواع فى لغات البرمجة حتى يستطيع المترجم التعرف على البيانات وإجراء العمليات المطلوبة عليها وحجز المساحة المناسبة لها فى الذاكرة حتى لا يتم إهدار المساحة المتوفرة فى الذاكرة بسرعة وتوقف عمل البرنامج أو توقف النظام بالكامل.

توجد فى فيجوال بيزك العديد من أنواع البيانات ، فالأعداد لها أنواع تخزن بها قيم صحيحة وأنواع تخزن بها قيم كسرية ، وكذلك توجد أنواع بيانات نصية ومنطقية وتاريخ ووقت وغير ذلك الكثير.

أنواع البيانات الرقمية الصحيحة فى فيجوال بيزك:

النوع	الحجم بالبايت	القيمة الصغرى	القيمة العظمى
Byte	١	٠	٢٥٥
SByte	١	١٢٨-	١٢٧
Short	٢	٣٢٧٦٨-	٣٢٧٦٧
UShort	٢	٠	٦٥٥٣٥
Integer	٤	٢١٤٧٤٨٣٦٤٨-	٢١٤٧٤٨٣٦٤٧
UInteger	٤	٠	٤٢٩٤٩٦٧٢٩٥
Long	٨	٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٨-	٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٧
ULong	٨	٠	١٨٤٤٦٧٤٤٠٧٣٧٠٩٥٥١٦١٥

النوع	الشرح
Byte	صحيح مختصر
Sbyte	صحيح مختصر قصير
Short	صحيح قصير
UnSort	صحيح قصير موجب
Integer	صحيح
UInteger	صحيح موجب
Long	صحيح طويل جداً
ULong	صحيح طويل جداً موجب

أنواع البيانات الرقمية العشرية :

النوع	الحجم بالبايت	القيمة الصغرى	القيمة العظمى
Single	٤	-3.4028235E38	3.4028235E38
Double	٨	1.79769313486231E308-	1.79769313486231E308
Decimal	١٦	يستخدم لتمثيل الأعداد الصحيحة والعشرية ويمكن تحديد الدقة العشرية من صفر إلى ٢٨ خانة عشرية	

أنواع البيانات النصية :

النوع	الحجم بالبايت	القيمة
Char	٢	يخزن به حرف واحد بترميز Unicode
String	غير محدود	يخزن به سلسلة حرفية من صفر إلى ٢ مليار حرف بترميز Unicode

النوع	الشرح
Single	حقيقي ذو دقة عالية
Double	حقيقي ذو دقة مضاعفة
Decimal	حقيقي ذو فاصلة ثابتة
Char	حرف
String	مجموعة حروف

كما يوجد أيضا أنواع أخرى من البيانات أهمها Boolean ويخزن به قيمة منطقية True/False ونوع Date ويخزن به التاريخ والوقت

كيف تحدد النوع المناسب للبيانات التى ستستخدمها فى برنامجك؟

بناء على تحديد نوع البيان يقوم المترجم بحجز المساحة اللازمة لتخزينه فى الذاكرة ، ويجب مراعاة تحديد النوع المناسب حسب القيمة التى سيتم التعامل معها فمثلا إذا كان المطلوب هو إدخال قيمة رقمية لدرجة طالب فى امتحان مادة اللغة العربية فكيف يتم اختيار النوع المناسب لهذه القمية ، ببساطة اتبع الخطوات التالية:

حدد أدنى قيمة يمكن إدخالها فى درجة اللغة العربية (صفر)

حدد أكبر قيمة يمكن إدخالها فى درجة اللغة العربية (٤٠)

حدد ما إذا كانت القيمة المطلوب إدخالها صحيحة أم عشرية (قيمة صحيحة)

إذن فالمطلوب هو حجز مساحة فى الذاكرة لتخزين عدد صحيح يقبل إدخال الأعداد من صفر إلى ٤٠

بالنظر إلى جدول أنواع البيانات الرقمية الصحيحة تجد أن النوع الأنسب فى هذه الحالة هو Byte أو SByte

لماذا لا نختار النوع Short أو Integer مثلا أو أى نوع صحيح آخر؟

لأن هذه الأنواع تحجز مساحة أكبر فى الذاكرة والقيمة المطلوب إدخالها لن تتعدى الرقم ٤٠ فليس هناك فائدة من تحديد نوع بيان أكبر بل بالعكس سيؤدى ذلك إلى إهدار المساحة المتوفرة فى الذاكرة وإبطاء تنفيذ العمليات التى ستتم على هذا البيان بدون داعى.

المتغيرات Variables

المتغير هو مكان يُحجز فى ذاكرة الحاسب لتخزين قيمة معينة هذه القيمة يمكن لها أن تتغير أثناء تشغيل البرنامج ، وهذا المتغير له اسم ونوع وحجم وقيمة.

قواعد تسمية المتغيرات:

- أن يبدأ اسم المتغير بحرف أبجدى أو علامة _ (الشرطة التحتية)
- يمكن أن يحتوى اسم المتغير على أرقام أو حروف أو علامة _
- عدم استخدام الكلمات المحجوزة للغة Keywords
- عدم استخدام أى علامات غير علامة الشرطة التحتية _
- عدم استخدام المسافات داخل اسم المتغير
- لا يجب أن يتعدى اسم المتغير ٢٥٥ حرف

تحديد نوع المتغير:

يتم تحديد نوع المتغير من الأنواع التى تم عرضها فى الجداول السابقة أو من الأنواع الأخرى الموجودة فى لغة فيجوال بيزك دوت نت ، ويراعى عند تحديد نوع المتغير أن يكون النوع مناسب للقيمة التى سيتم إسنادها إليه وأن يكون المدى المسموح به لهذا النوع مناسب لأقصى قيمة سيتم تخزينها فيه.

مثال:

إذا أردت كتابة برنامج لحساب النسبة المئوية لمجموع طالب فى المرحلة الثانوية ، فما هى المدخلات والمخرجات الخاصة بالبرنامج وأنواع البيانات المناسبة لكل منها

مدخلات البرنامج ستكون (درجة الطالب – المجموع الكلى)

مخرجات البرنامج (النسبة المئوية)

أنواع البيانات المناسبة:

بالنسبة للمدخلات فدرجة الطالب ستكون رقم يبدأ من الصفر وينتهى عند المجموع الكلى ويمكن أن تكون الدرجة رقما صحيحا (٣٢٠) أو ربما تحتوى على جزء عشرى (٣٥٣٥) ، وبالنسبة للمجموع الكلى فسيكون رقما صحيحا ثابتا وليكن (٤١٠) فالنوع المناسب له هو **Short** أو **UShort** ودرجة الطالب ستكون من النوع **Single** لآنها يمكن أن تحتوى على جزء عشرى.

أما بالنسبة للمخرجات فستكون رقم يبدأ من الصفر وينتهى عند ١٠٠ وأيضا يمكن أن يتضمن جزء عشرى (٩٢٢٨) فالنوع المناسب له هو **Single**.

حجم المتغير (المدى المتاح له فى الذاكرة) :

يُقاس حجم المتغير بوحدة البايت **Byte** والتى تحتوى بدورها على عدد ٨ **Bit** والـ **Bit** هى أصغر وحدة تخزين فى الحاسب حيث يُخزن بها رقم (١) أو (٠) بالنظام الثنائى (لغة الآلة)

قيمة المتغير :

هى القيمة التى تُسند إليه والتى يمكن أن تتغير أثناء تشغيل البرنامج ويجب أن تقع هذه القيمة فى المدى المسموح به لنوع المتغير فمثلا متغير من نوع **SByte** من الخطأ أن تُسند له قيمة مثل ١٩٠ لآن أقصى قيمة مسموح بها لهذا النوع هى ١٢٧

الإعلان عن المتغير : Declaring Variable

يجب الإعلان عن أى متغير قبل استخدامه وعملية الإعلان تتضمن إنشاء المتغير وتحديد اسمه ونوعه وبناء على النوع المحدد للمتغير يتم حجز المساحة اللازمة فى الذاكرة

صيغة الإعلان عن المتغير:

```
Dim [Variable Name] As [Data Type]
```

تستخدم الكلمة المحجوزة Dim للإعلان عن المتغير

يتم تحديد اسم المتغير Variable Name وفق قواعد تسمية المتغيرات

ثم تكتب الكلمة المحجوزة As

ثم يتم تحديد نوع المتغير Data Type

مثال:

```
Dim Student_Mark As Short
Dim FirstName As String
Dim Status4 As Boolean
```

ويمكن الإعلان عن أكثر من متغير من نفس النوع بهذا الشكل :

```
Dim A, B, C As Byte
```

إسناد قيمة للمتغير Assignment :

يتم إسناد القيم للمتغيرات بهذا الشكل :

```
[Variable Name] = [Value]
```

يُكتب اسم المتغير ثم علامة الإسناد = ثم القيمة

والقيمة يمكن أن تكون قيمة مباشرة مثل :

```
Student_Mark = 120
```

ويمكن أن تكون تعبير Expression مثل :

```
Student_Mark = 9 * 6
```

ويمكن أن تكون قيمة من متغير آخر مثل :

```
Student_Mark = C
```

ويمكن إسناد قيمة للمتغير أثناء الإعلان عنه :

```
Dim Student_Mark As Integer = 180
```

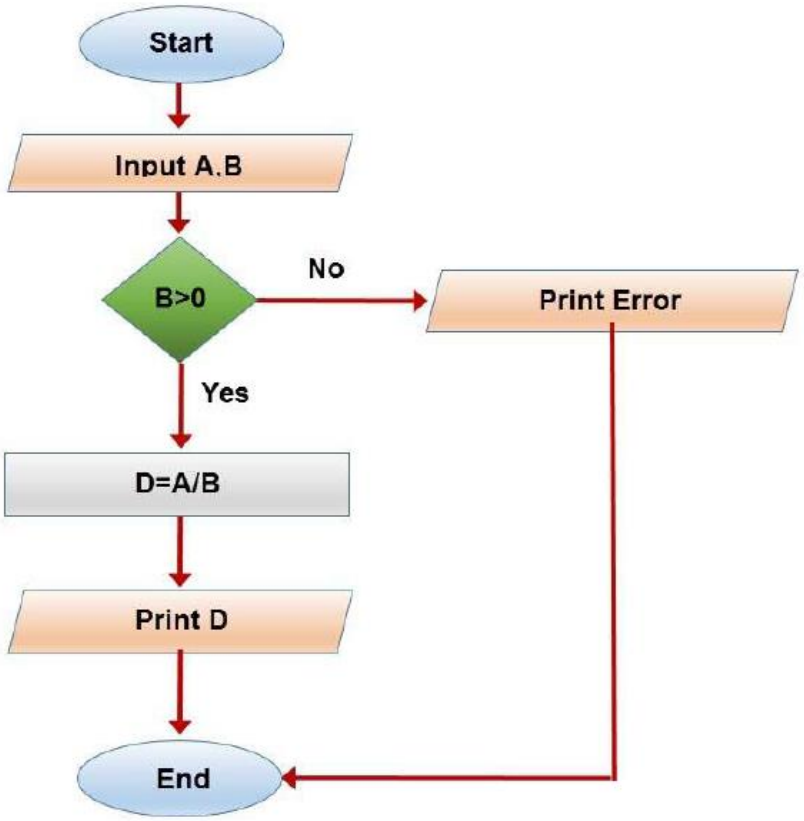
عودة إلى الجزء العملى:

المثال الثانى:

ارسم خريطة التدفق التى توضح سير خطوات برنامج يقوم بحساب ناتج فسمة عددين؟

فى هذا المثال سنحتاج إلى إتخاذ قرار مشروط والذى يتم تمثيله فى خرائط التدفق بشكل المُعين ، حيث أنه يجب علينا اختبار قيمة العدد الثانى الذى سيأتى فى المقام والذى يجب أن يكون أكبر من الصفر لآن القسمة على صفر ليس لها معنى.

نقوم برسم خريطة التدفق بالشكل التالى:



بعد إدخال قيم المدخلات A,B يتم اختبار قيمة العدد الثانى إن كان أكبر من الصفر يتم إجراء عملية القسمة وإظهار الناتج ثم إنهاء البرنامج وإن لم يتحقق هذا الشرط فلا يجب أن تتم عملية القسمة لذا يتم تحويل مسار البرنامج إلى خطوة أخرى وهى طباعة رسالة خطأ ثم إنهاء البرنامج.

الكود

```
Module Module3
    Sub Main()
        Dim A, B As Integer
        Dim V As Single
        Console.WriteLine("Enter tow numbers:")
        A = Console.ReadLine()
        B = Console.ReadLine()
        If B > 0 Then
            V = A / B
        Else
            Console.WriteLine("error:")
        End If
        Console.WriteLine("the divisson is:")
        Console.WriteLine(V)
        Console.ReadKey()
    End Sub
End Module
```

إنشاء مشروع جديد New project

من قائمة File اختر new project

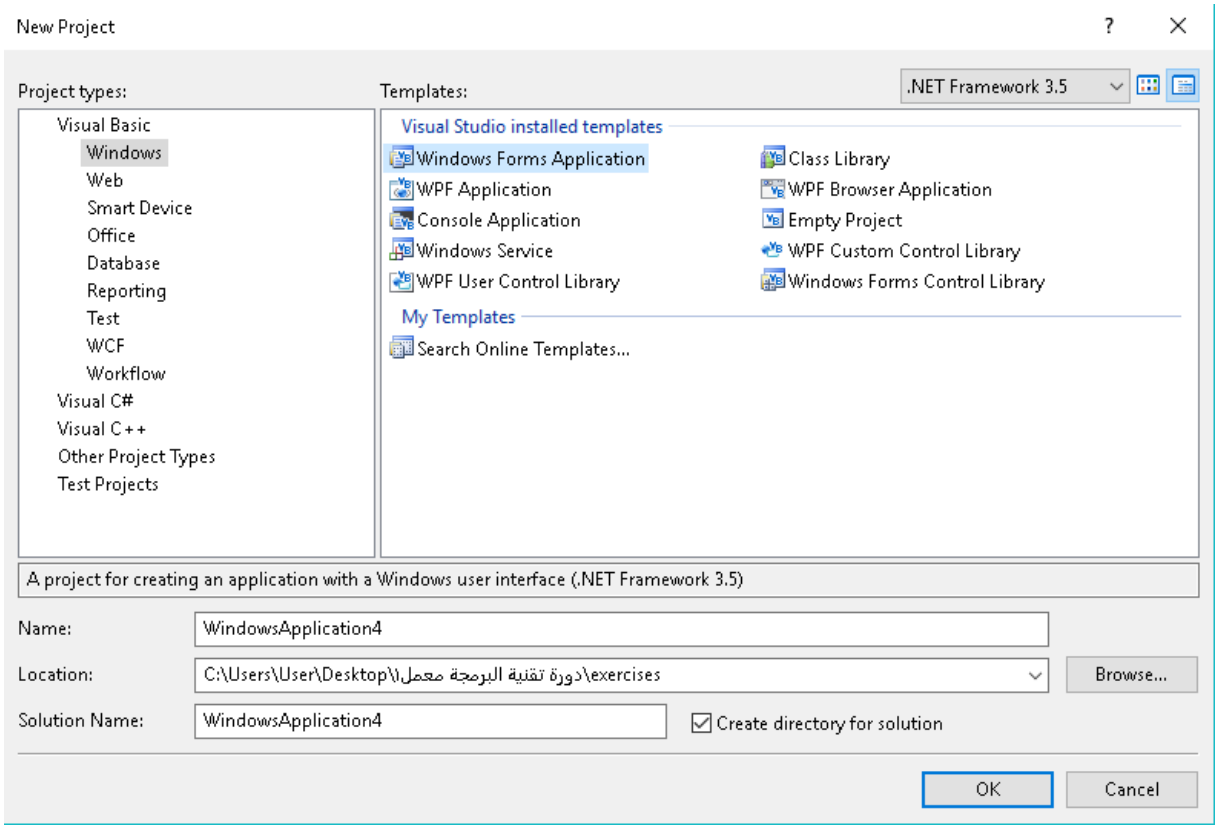
في مربع مشروع جديد New project نختار Visual Basic

نختار ويندوز في Project Type

نختار Windows Application من Templates

ندخل اسم المشروع في المكان المخصص Name ومكان الحفظ

نضغط OK



منهجية بناء المشروع بلغة VB.net :

أي برنامج أو مشروع مكتوب بلغة VB.net يحتوي على خمسة عناصر هي :

1. **النماذج Form**

تمثل واجهات المشروع والوعاء الذي يحتوي على جميع أغراضه

2. **أدوات التحكم Control Tools**

وهي عناصر التحكم التي يتكون منها المشروع

3. **الخصائص Properties**

وهي الصفات التي تتمتع بها الأداة أو الكائن

4. **الأحداث Events**

وهي المؤثرات التي تطرأ على المشروع من قبل الفأرة أو لوحة المفاتيح أو غير ذلك وينتج **من ورائها تنفيذ التعليمات البرمجية.**

5. **التعليمات Statements (الشيفرة Code)**

هي مجموعة الأوامر التي تنفذ عند اختيار حدث ما، وينتج من وراء ذلك عمل معين.

دراسة الأدوات :

- **زر الأمر Button** : يستخدم لتنفيذ مجموعة تعليمات عند اختيار حدث معين
خصائصه:

اسم الخاصية	الوظيفة
Text	يظهر النص على الزر
Enabled	تفعيل وتعطيل الزر وتأخذ القيمة True/False
Font	تنسيق الخط الذي يظهر على الزر
Forecolor	لون الخط لنص الزر
Backcolor	لون خلفية الزر
Visible	التحكم في اظهار واخفاء الزر
Right to left	تغيير اتجاه النص من اليمين الى اليسار والعكس

يتم ضبط الخصائص بطريقتين :

- **أثناء التصميم** : يتم ضبط الخصائص أثناء تصميم المشروع باستخدام نافذة الخصائص
- **اثناء التنفيذ** : هناك بعض الخصائص لا تكون متاحة الا عند تشغيل المشروع ولتغيير الخصائص أثناء تشغيل المشروع نتبع الصيغة الآتية :

Control_name.Property_name = New_Value

أمثلة :

```
Button1.Visible= false
TextBox1.MultiLine = True
Label1.ForeColor =Color.Red
Form1.BackColor=Color.Yellow
```

- **أداة النص Text Box** : تستخدم لإدخال البيانات واخراج النتائج
خصائصها :
جميع خصائص زر الأمر Button أعلاه تشترك فيها أداة النص Text Box
- **أداة العنوان Label** : تستخدم لعرض النصوص الثابتة التي لا يستطيع المستخدم تعديلها
جميع خصائص زر الأمر Button أعلاه تشترك فيها أداة العنوان Label

خصائص النموذج Form :

اسم الخاصية	الوظيفة
Text	تغيير نص شريط العنوان للنموذج
BackColor	تغيير لون الخلفية
MaximizeBox	تحدد امكانية ظهور زر التكبير في شريط العنوان
MinimizeBox	تحدد امكانية ظهور زر التصغير في شريط العنوان
Right to left	تغيير اتجاه النص من اليمين الى اليسار والعكس في شريط العنوان

مثال:

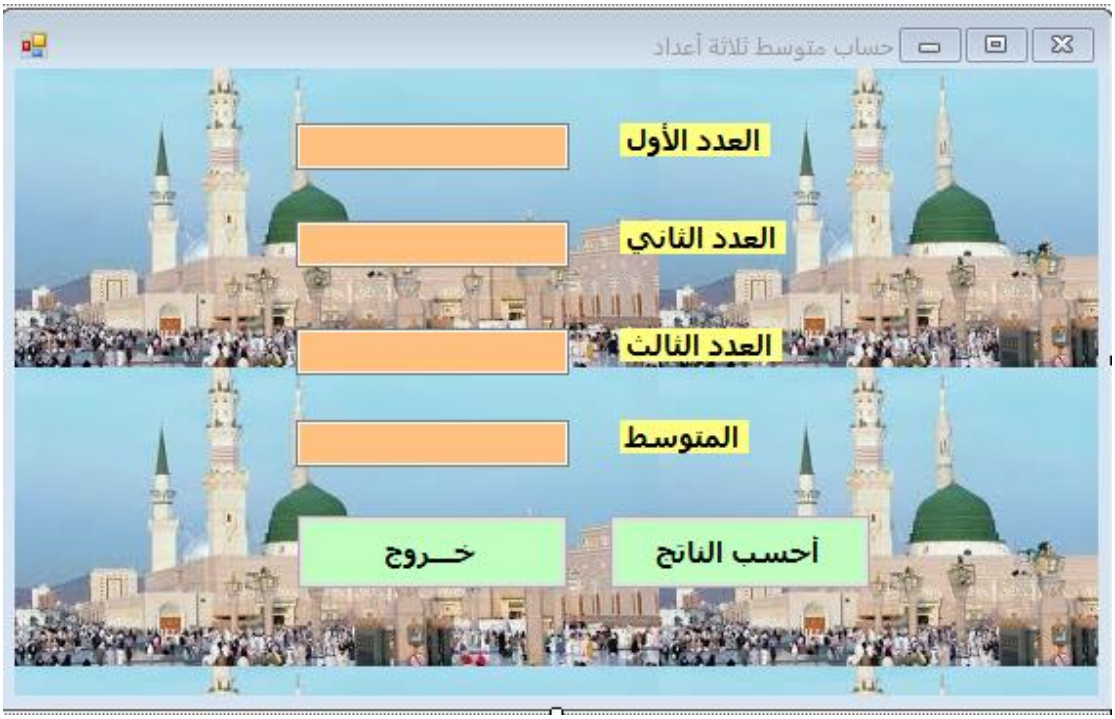
تصميم برنامج التحية ورد التحية في الاسلام

أولاً: تصميم الشاشة


```
Public Class Form1

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button1.Click
        TextBox1.Text = "وبـركـاتـه تـعـالـى الله ورحمة عليكم الـسلام"
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button2.Click

        TextBox2.Text = " وبـركـاتـه تـعـالـى الله ورحمة الـسلام وعليكم "
    End Sub

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        TextBox1.Text = "  "
        TextBox2.Text = "  "
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Button4.Click
        End
    End Sub

End Class
```

صمم برنامج لإيجاد متوسط ثلاثة أعداد

التصميم



الكود

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        Dim a, b, c As Integer
        Dim d As Single
        a = TextBox1.Text
        b = TextBox2.Text
        c = TextBox3.Text
        d = (a + b + c) / 3
        TextBox4.Text = d
    End Sub
```

هياكل القرارات:

مقدمة

فى جميع البرامج تقريبا نحتاج إلى اتخاذ بعض القرارات بناء على شروط معينة وكذلك نحتاج أحياناً إلى تكرار بعض الأوامر لعدد معين من المرات بدلاً من إعادة كتابتها أكثر من مرة ، توفر لغات البرمجة ما يسمى بهياكل القرارات أو التفرع والتى تُستخدم فى تنفيذ أوامر معينة عندما يتحقق شرط معين ، وتوفر أيضاً ما يسمى بهياكل التكرار والتى تستخدم فى تكرار أوامر معينة لعدد معين من المرات.

هياكل القرارات :

تقدم لنا لغة فيجوال بيزك دوت نت مجموعة من التعليمات التى تقوم بتنفيذ تعليمة أو مجموعة من التعليمات بناء على قيمة منطقية لشرط معين ، والقيمة المنطقية هى إما True أو False والشرط هو أى تعبير له قيمة منطقية فيمكن أن يكون الشرط ناتج مقارنة بين متغيرين باستخدام عوامل المقارنة مثل A>b فهذا يسمى تعبير شرطى تكون نتيجته قيمة منطقية إما True إذا تحقق الشرط وإما False إذا لم يتحقق الشرط ، وتعليمات إتخاذ القرار فى فيجوال بيزك دوت نت هى :

- If ..... Then
- If ..... Then ..... Else
- Select Case

النوع الأول:

تعليمة If ..... Then

فى حالة تنفيذ سطر واحد فقط من الكود تُكتب جملة If على الصورة التالية :

```
If  [Conditional Exprision]  Then [Statement]
```

تُكتب الجملة على سطر واحد بحيث تبدأ بالكلمة المحجوزة If ثم يأتى بعدها تعبير شرطى ثم يليه الكلمة المحجوزة Then ثم الجملة المطلوب تنفيذها.

النوع الثاني:

تعليمة If ..... Then ..... Else

تُستخدم هذه الجملة إذا أردت تنفيذ تعليمات معينة فى حالة عدم تحقق الشرط ، وتُكتب هذه التعليمات بعد الكلمة المحجوزة Else

مثال للنوع الثاني:

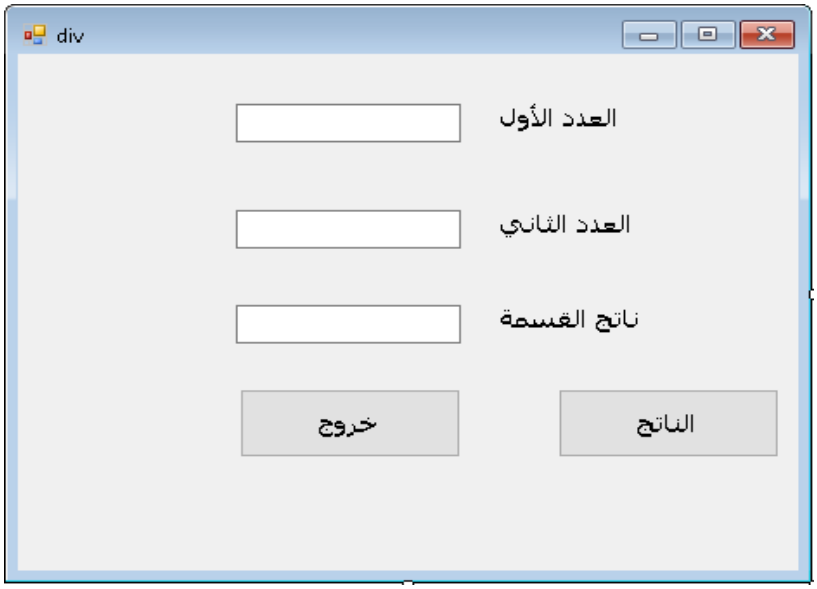
عودة لمثال سابق تم عمل خريطة تدفق له

صمم برنامج لإيجاد ناتج قسمة عددين

تصميم الشاشة المطلوبة

بإضافة الأدوات وتغيير بعض خصائصها وهي

Button ،Label ،Textbox

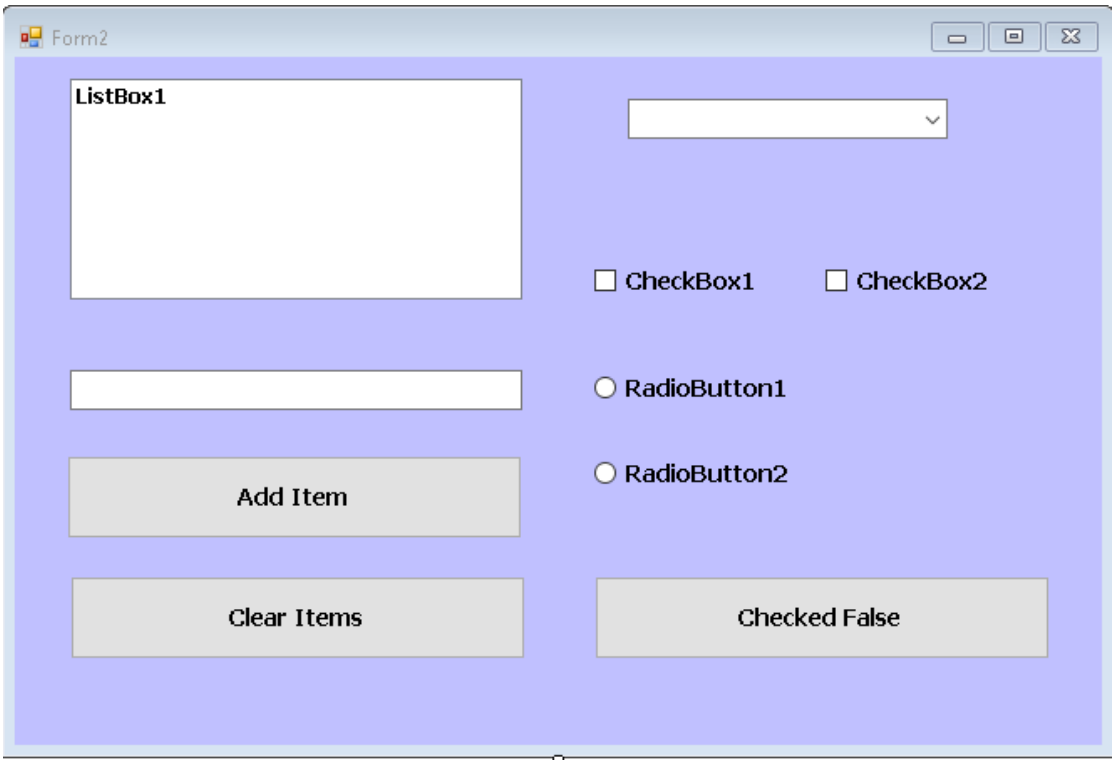


الكود على زر الناتج

```
Public Class div

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim a, b As Integer
        Dim d As Single
        a = TextBox1.Text
        b = TextBox2.Text
        If b > 0 Then
            d = a / b
        Else
            MsgBox("Error no result", MsgBoxStyle.OkOnly, "خطأ")

        End If

        TextBox3.Text = d
    End Sub
```

المزيد من أدوات التحكم:

صمم النافذة كما بالشكل التالي:



الكود

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        ListBox1.Items.Add(TextBox1.Text)
        ComboBox1.Items.Add(TextBox1.Text)
        TextBox1.Text = ""

    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        CheckBox1.Checked = False
        CheckBox2.Checked = False
        RadioButton1.Checked = False
        RadioButton2.Checked = False
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        ListBox1.Items.Clear()
        ComboBox1.Items.Clear()
    End Sub
```

```
    Private Sub Form2_Load(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles MyBase.Load
        CheckBox1.Checked = True
        RadioButton1.Checked = True

    End Sub
```

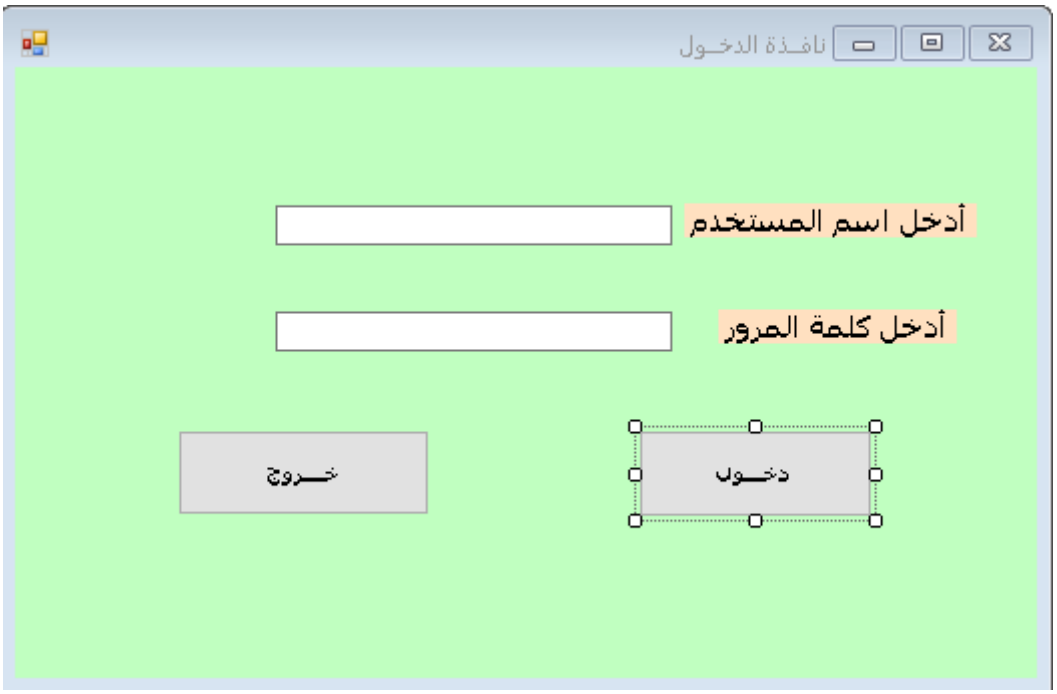
إضافة نافذة جديدة للمشروع

لإضافة نافذة جديدة للمشروع Form2 أنقر قائمة Project ثم اختر Add Windows Form

لبداية تنفيذ البرنامج من نافذة Form2 أنقر قائمة Project ثم اختر WivdowsApplication Properties

صمم برنامج لتسجيل دخول مستخدم

تصميم النافذة



الكود

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        If TextBox1.Text = "mohamed" And TextBox2.Text =
"adam" Then
            MsgBox("الدخول يمكنك الآن بك مرحبا")
Form1.showdialog()
        Else
            MsgBox("الدخول يمكنك لا نأسف")
        End If
    End Sub
```

تغيير الخصائص برمجياً:

الكود:

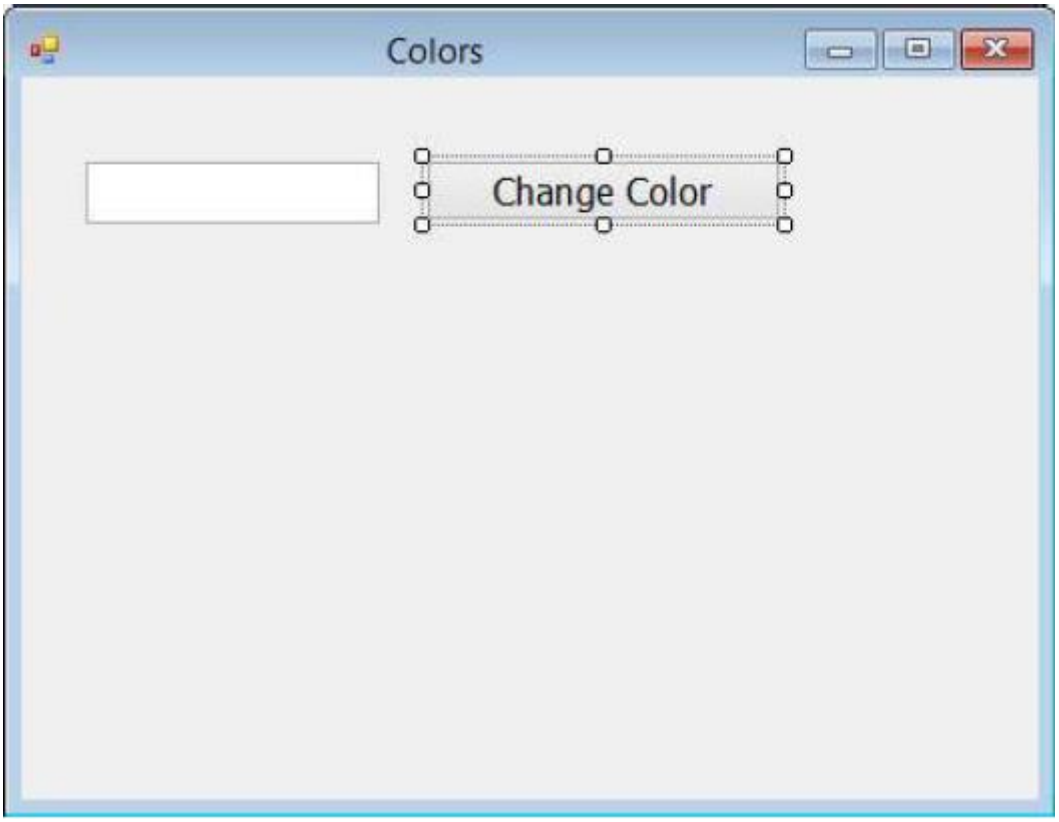
```
Public Class Form5

    Private Sub Button1_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
Button1.Click
        Me.BackColor = Color.Red
    End Sub
```

```
    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        Me.BackColor = Color.Blue
    End Sub
    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click
        Me.BackColor = Color.Green
    End Sub
    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        Me.BackColor = Color.Yellow
    End Sub
End Class
```

مثال2:

صمم برنامج لتغيير لون خلفية النافذة حسب اسم اللون المدخل في النص

التصميم



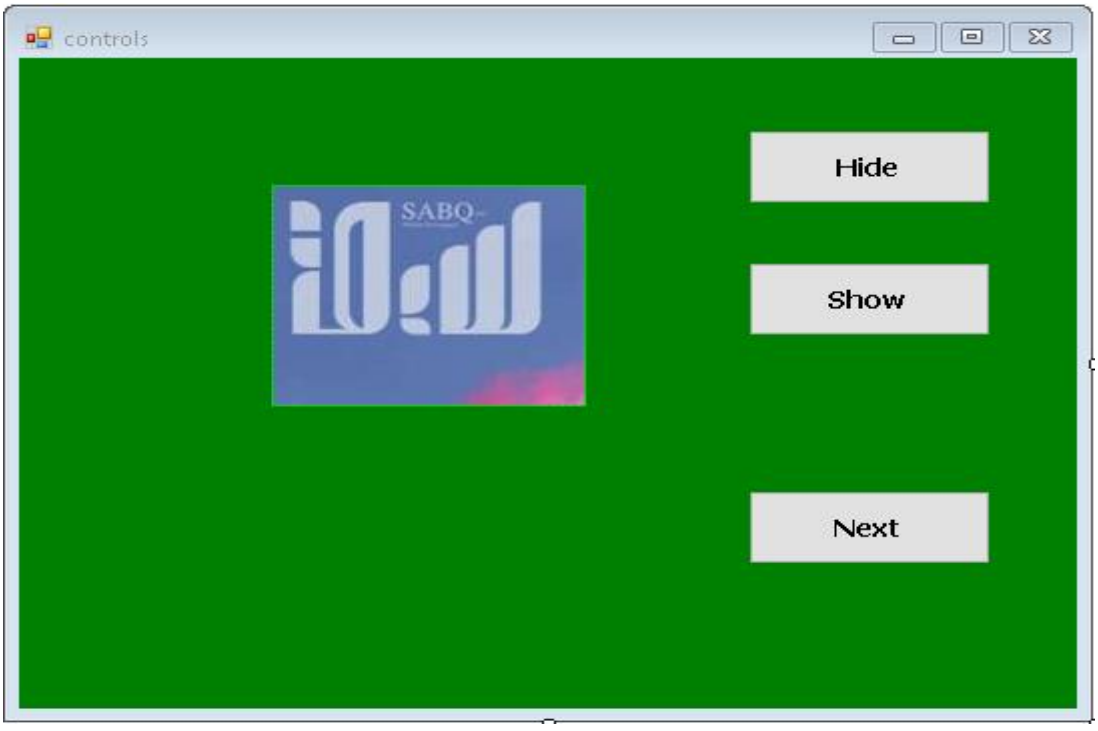
وغير خصائص الأدوات كما يلى :

الأداة	الخاصية	القيمة
Button	Name	btnColor
Button	Text	Change Color
TextBox	Name	txtColor
Form	Text	Colors

اضغط على الزر btnColor ضغطة مزدوجة بالفأرة لفتح نافذة الكود واكتب الكود التالى فى الحدث Click

```
Public Class Form1

    Private Sub btnColor_Click(sender As Object, e As EventArgs)
        If txtColor.Text = "Red" Then
            Me.BackColor = Color.Red
        ElseIf txtColor.Text = "Blue" Then
            Me.BackColor = Color.Blue
        ElseIf txtColor.Text = "Green" Then
            Me.BackColor = Color.Green
        End If
    End Sub
End Class
```

أضف نافذة أخرى للمشروع كالتالي:



الكود

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        PictureBox1.Hide()
    End Sub

    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        PictureBox1.Show()
    End Sub

    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        Form2.ShowDialog()
    End Sub
```

تكملة لبنيات اتخاذ القرار

جملة (بنية القرار: select case)
الصيغة العامة

```
select case variable
        case value1
                statement
        case value2
                statement
        case value3
                statement
        case else
                statement
end select
```

مثال:

صمم برنامج آلة حاسبة بسيطة لإجراء عمليات الجمع والطرح والضرب والقسمة لعددين باستخدام بنية القرار select case

التصميم:

الكود

```
Dim num1, num2 As Integer
        Dim operat As String
        num1 = TextBox1.Text
        operat = TextBox2.Text
        num2 = TextBox3.Text
        Select Case operat
            Case "+"
                TextBox4.Text = num1 + num2
            Case "-"
                TextBox4.Text = num1 - num2
            Case "*"
                TextBox4.Text = num1 * num2
            Case "/"
                TextBox4.Text = num1 / num2
          Case eles
          Msgbo("error")
        End Select
```

استعمال الحلقات

1. الحلقة For....Next

تتيح هذه الحلقة تنفيذ مجموعة معينة من الجمل البرمجية عددا محددا من المرات

الصيغة العامة لها

for variable = start **to end**

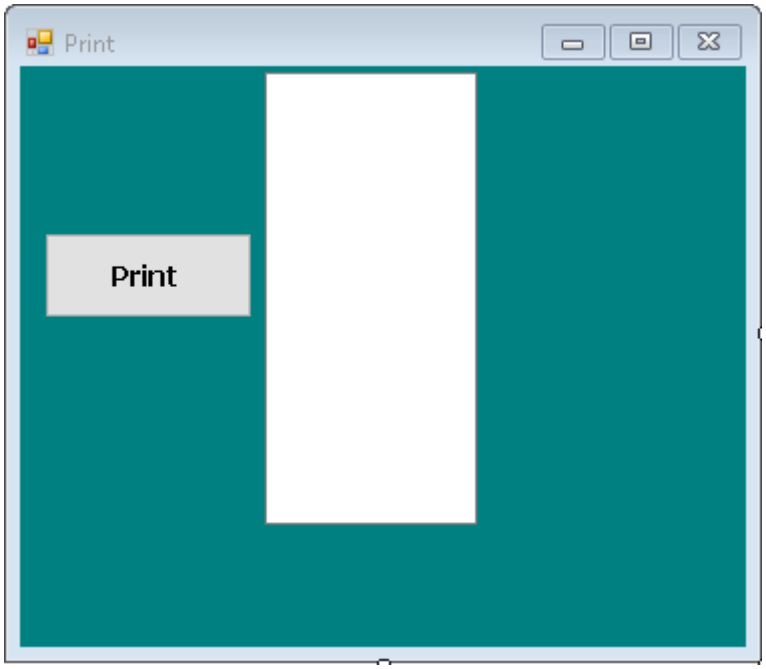
statements to be repeated

next variable

مثال1:

أكتب برنامج لطباعة نص VB.NET مقدار عشرة سطرية مرات داخل مربع نص

تصميم النافذة



يجب تغيير خاصية Multi Line لأداة مربع النص إلى True

الكود:

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        Dim j As Integer
        Dim x As String
        x = Chr(13) + Chr(10)

        For j = 1 To 10
            TextBox1.Text = TextBox1.Text & "VB.NET" & x
        Next
    End Sub
```

الأداة & تستخدم للدمج والجملة Chr(13) + Chr(10) وظيفتها نقل مؤشر الفأرة لسطر جديد

قم بالتعديل على البرنامج بحيث يستقبل من المستخدم عدد مرات طباعة الكلمة VB.NET

الكود:

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        Dim j,i As Integer
        Dim x As String
        x = Chr(13) + Chr(10)
        i = InputBox("الطباعة مرات عدد أدخل")
        For j = 1 To i
            TextBox1.Text = TextBox1.Text & "VB.NET" & x
        Next
    End Sub
```

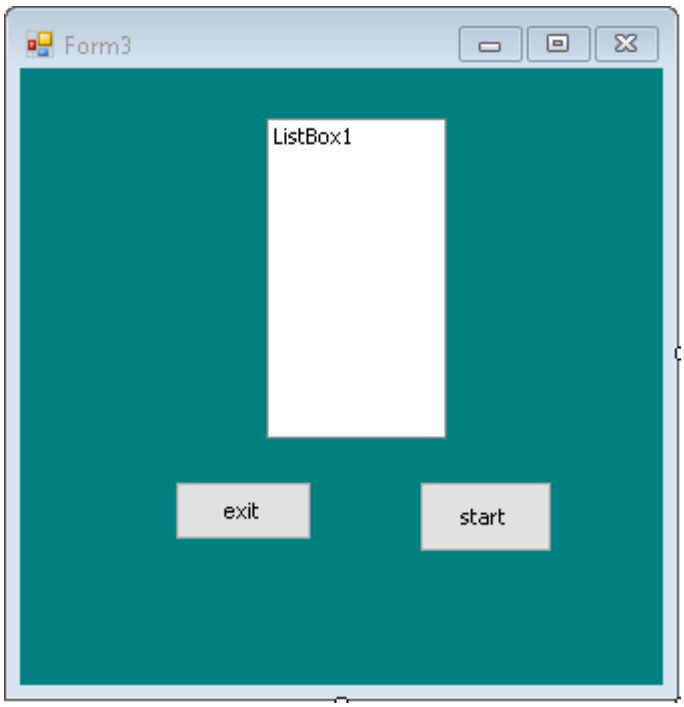
قم بالتعديل على الكود:

```
Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        Dim j,i As Integer
        Dim x As String
        x = Chr(13) + Chr(10)
        i = InputBox("الطباعة مرات عدد أدخل")
        For j = 1 To i
            TextBox1.Text = TextBox1.Text & j & x
        Next
    End Sub
```

مثال2 على حلقات التكرار:

برنامج لطباعة حاصل ضرب عددين حسب الأعداد التي يتم إدخالها باستخدام حلقات التكرار حيث العدد الأول يمثل عدد تكرار الحلقة والعدد الثاني المضروب فيه أعداد الحلقة

تصميم النافذة



الكود

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles Button1.Click
        Dim i, j, x As Integer
        x = InputBox("enter loop number")
        i = InputBox("enter number2")
        For j = 1 To x
            ListBox1.Items.Add(j & "*" & i & "=" & j * i)
        Next
    End Sub
```

عودة للحلقات التكرارية

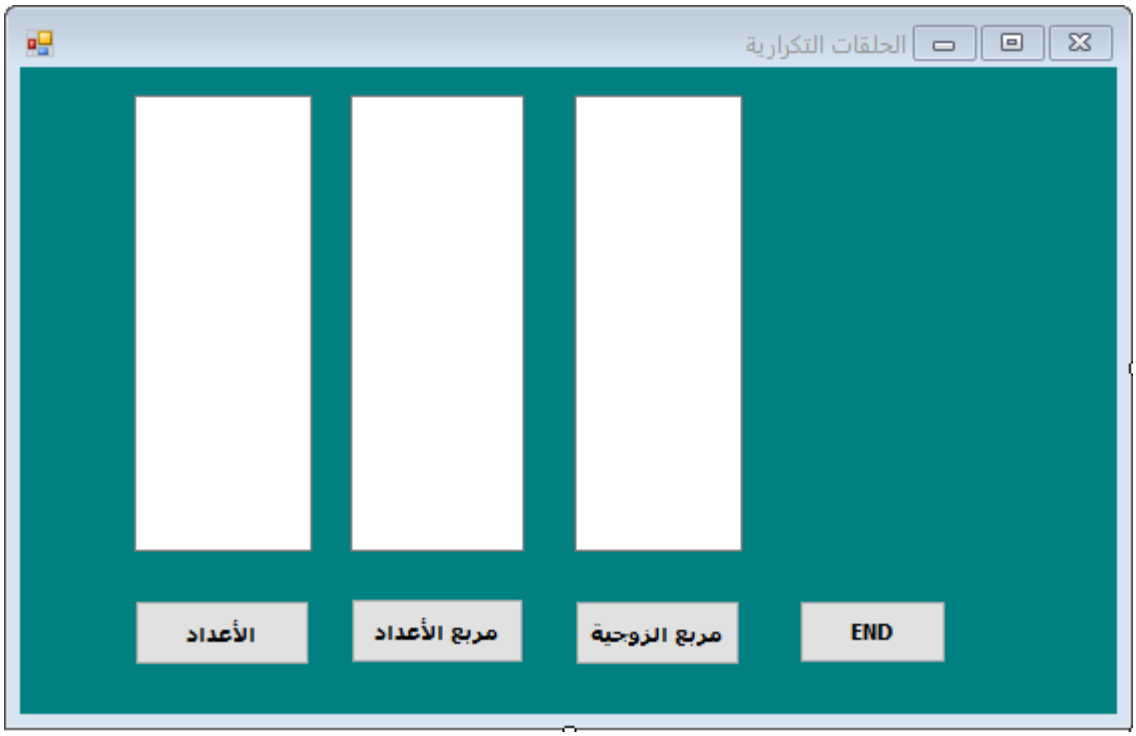
مثال4: باستخدام الحلقة التكرارية for .....next صمم برنامج لعرض

أرقام من 1 الى رقم مدخل من المستخدم

وإيجاد مربع العدد

ثم مربع الأعداد الزوجية من 2 الى ذلك الرقم

التصميم



الكود

```
Public Class Form4
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
        Dim i, j As Integer
        Dim y As String
        y = Chr(13) + Chr(10)
        j = InputBox("enter number")
        TextBox1.BackColor = Color.Yellow
        TextBox1.ForeColor = Color.Blue
        For i = 1 To j
            TextBox1.Text = TextBox1.Text & y & i
        Next
    End Sub
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
        Dim i, j As Integer
        Dim y As String
        y = Chr(13) + Chr(10)
        j = InputBox("enter number")
```

```
        TextBox2.BackColor = Color.Silver
        TextBox2.ForeColor = Color.Red
        For i = 2 To j
            TextBox2.Text = TextBox2.Text & y & i ^ 2
        Next
    End Sub
    Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
        Dim i, j As Integer
        Dim y As String
        y = Chr(13) + Chr(10)
        j = InputBox("enter number")
        TextBox3.BackColor = Color.Pink
        TextBox3.ForeColor = Color.Green
        For i = 2 To j Step 2
            TextBox3.Text = TextBox3.Text & y & i ^ 2
        Next
    End Sub
    Private Sub TextBox3_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles TextBox3.TextChanged
    End Sub
    Private Sub Button1_MouseMove(ByVal sender As Object, ByVal e As
System.Windows.Forms.MouseEventArgs) Handles Button1.MouseMove
        TextBox1.BackColor = Color.Red
    End Sub
End Class
```

الحلقات التكرارية المتداخلة:

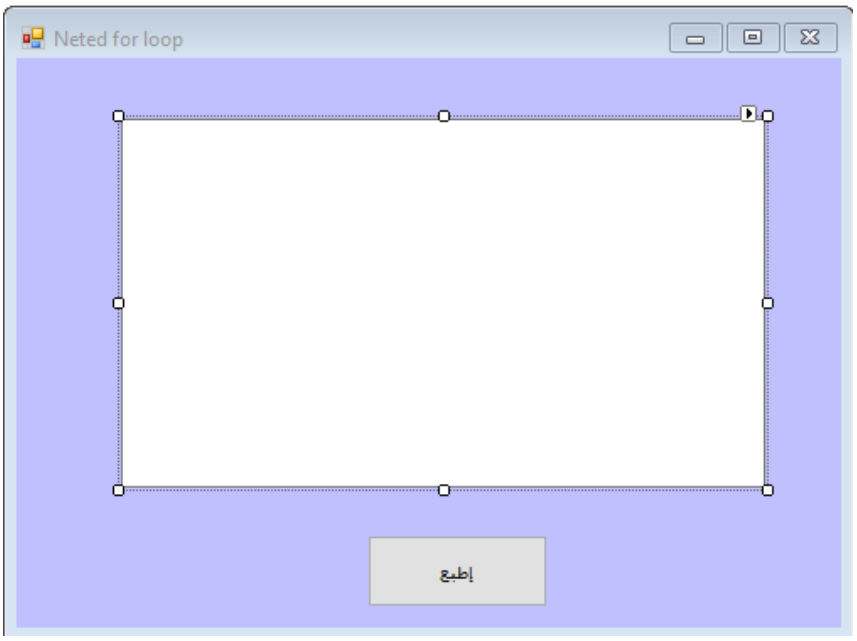
في هذه تكون هنالك حلقة داخل حلقة

بحيث تأخذ الحلقة الداخلية كل قيمها بينما تأخذ الحلقة الخارجية القيمة الواحدة

مثال: صمم برنامج لطباعة الأعداد التالية في مربع نص

```
1
2    2
3    3    3
4    4    4    4
5    5    5    5    5
```

التصميم:



الكود:

```
Dim i, j As Integer
        Dim x As String
        x = Chr(10) + Chr(13)
        For i = 1 To 5
            For j = 1 To i
                TextBox1.Text = TextBox1.Text & i & Chr(9) & x
            Next
            TextBox1.Text = TextBox1.Text & x
        Next
```

المصفوفات Arrays

هي عبارة عن تنظيم للعناصر في صورة صفوف و أعمدة من العناصر المحددة والمتجانسة من نوع واحد يتم تخزينها في منطقة من الذاكرة باسم واحد، وبذلك يكون من السهل الوصول لأي عنصر من عناصر المصفوفة من خلال موقعه فيها.

تنقسم المصفوفات إلى نوعين:

- مصفوفة ذات بعد واحد.
- مصفوفة ذات بعدين

1- المصفوفات ذات البعد الواحد (One Dimensional Array)

الإعلان عن المصفوفة:

الصيغة العامة للإعلان

```
Dim a(9) As Integer
```

وهو عبارة عن تصريح عن مصفوفة من 10 عناصر صحيحة

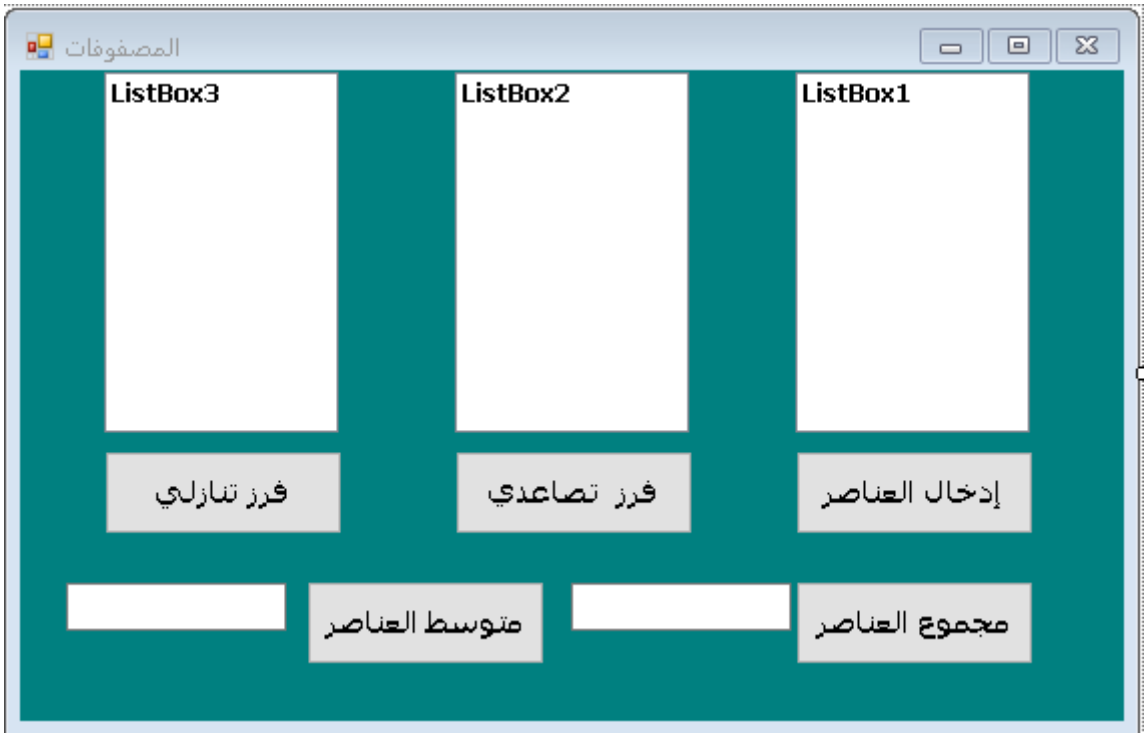
بعض الدوال التي تتعامل مع المصفوفات

1. االدالة UBound(a): الحد العلوي للمصفوفة
2. الدالة LBound(a): الحد السفلي للمصفوفة
3. Array.Sort : فرز المصفوفة تصاعديا
4. Array.Reverse : فرز المصفوفة تنازليا

مثال:

أكتب برنامج لقراءة عناصر مصفوفة ذات بعد واحد تتكون من 6 عناصر ثم إيجاد مجموع العناصر و إيجاد الوسط الحسابي للعناصر.

التصميم:



الكود البرمجي:

```
Public Class arrays
    Dim a(5) As Integer
    Dim i As Integer
    Private Sub Button1_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button1.Click
        For i = 0 To 5
            a(i) = InputBox("enter number" & i + 1)
            ListBox1.Items.Add(a(i))
        Next
    End Sub
    Private Sub Button2_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button2.Click
        Array.Sort(a)
        For i = 0 To 5
            ListBox2.Items.Add(a(i))
        Next
    End Sub
    Private Sub Button3_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button3.Click
        Array.Reverse(a)
        For i = 0 To 5
            ListBox3.Items.Add(a(i))
        Next
    End Sub

    Private Sub Button5_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button5.Click
```

```
        Dim sum As Integer
        For i = 0 To 5
            sum = sum + a(i)
        Next
        TextBox1.Text = sum
    End Sub

    Private Sub Button4_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles Button4.Click
        Dim sum As Integer
        Dim ave As Single
        For i = 0 To 5
            sum = sum + a(i)
        Next
        ave = sum / 5
        TextBox2.Text = ave
    End Sub
End Class
```

المراجع:

1.خطوة على طريق فيجوال بيسك دوت نت، سامح السيد كامل

2. Lecture of Visual Basic 2008 إعداد المهندس: محمد أبو العلا