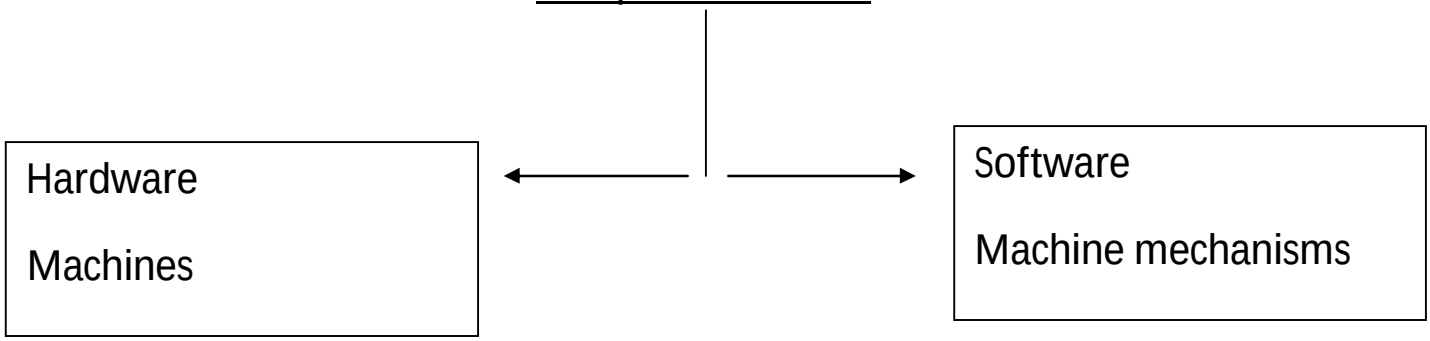
# Programming Principles

Introduction: -

## Computer Machine

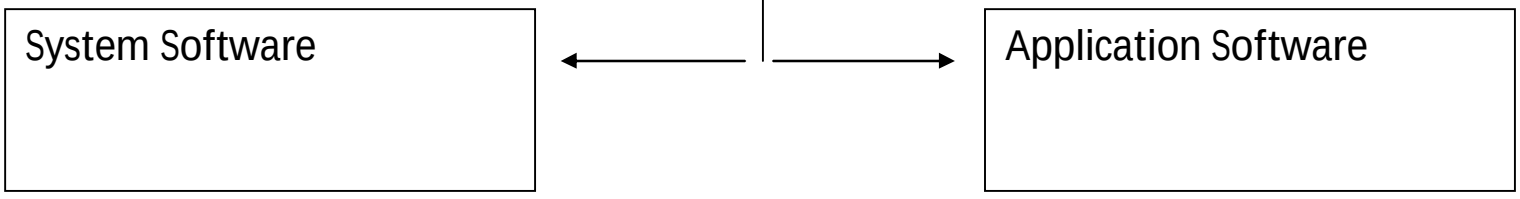



الجزء الاول وهو البرمجيات وهى التى تعطى الاجزاء الماديه فائدتها وبدونه لا جدوى لها حيث انه يعتبر المحرك الاساسى لها والذى يقوم بتنظيم حركة الهاردوير وترتيب والتحكم فى مهامه وطريقة اتصال اجزاء الهاردوير ببعضها البعض

الجزء الثانى وهى الاجزاء الماديه الاساسيه التى تصنع بأمكانيات محدده ولاغراض محدده

وبالتالى يتضح لنا ان المنظومه كاملة تتكون من جزئين لاغنى لاحدهما عن الاخر ولعمل المنظومه يجب ان تكتمل اولا ثم تبداء العمل فيها

## Software Type



## System Software

وهى انظمة التشغيل التى تقوم بالاعتماد عليها لتشغيل الاجزاء الماديه المكونه لجهازك الحاسوب وهى وسيلتك الاساسيه للتحدث مع جهازك او مع الاله التى تملكها

## Applications Software

وهى البرامج التى تحتاجها لاداء مهام معينه على جهازك والتى اما ان تكون برامج اساسيه والتى يقوم نظام التشغيل بتنزيلها على جهازك والتى تتبع جميعها الشركه المنتجه لنظام التشغيل وتتيح لك تنظيم العمل على جهازك وادارة ملفاتك باحترافيه وسهوله

او البرامج التى تقوم انت بشرائها من اى شركة منتجه لها او متداوله على مواقع الانترنت والتى تقوم باداء وظائف معقده فى مجالات الكمبيوتر المتنوعه وتسمى هذه البرامج

## Third party

ان نظام التشغيل قادر على ان يقوم بارسال اوامر البرامج المختلفه والتى يصنعها المبرمجون الى اجزاء الحاسوب الماديه والتى تقوم بدورها بتنفيذ تلك الاوامر بدقه واظهار نتائجها للمستخدم اذا تجد ان نظام التشغيل يلعب دور الوساطه بين المستخدم واجزاء الحاسوب الماديه كما انه يلعب دور الوسيط ايضا بين البرامج وبين الاجزاء الماديه

ما هى برامج الكمبيوتر : هى كتابة خطوات منطقيه لحل مشكله معينه والتى يتم تعليمها للاله لكى تقوم هى بحل تلك المشكله فيما بعد

# Languages

ان اللغات هى اداة تواصل بين المجتمعات المختلفه وهى لغه يفهمها البشر لكن اذا ما اردنا ان نتواصل مع الاله الاليكترونيه فانها لا تفهم لغتنا بل تفهم لغة 0 و 1 لكن هل سيفهم الانسان تلك اللغه

| Low level language | Assembly | High level language |
|---|---|---|
| هى لغة ال0 و1 وهى لغة لا يفهما الى الاله<br>وبالاخص يفهمها وحدة المعالجة المركزيه او البروسيسور ويعتبر هو العقل المدبر والمفكر لاى جهاز حاسوبى حيث انه بتعامل بالاشارات الكهربيه وهى 0 و1 فكل اشاره منها تعنى له شىء فى تنفيذ الاوامر<br><br>لذلك يجب ان ترسل اليه كافة الاوامر بهذه اللغه والا فلن يستطيع تنفيذ اى شىء<br><br>ويرد على تنفيذ الاوامر بنفس الطريقه وهنا ياتى دور المترجمات فى فهم المعالج وتفسير ما يقول للمستخدم | هذه اللغه التى يخلط الكثير بينها وبين لغة الاله لكن هى لغه يفهمها ايضا الانسان لانها تستخدم فيها الرموز انجليزيه<br>كما ان بعض لغات البرمجه تتم الكتابه فيها بلغة الاسيمبلى مثلا كبرامج المايكرو لكن ماهى فائتدها هى تعتبر خلفية لغة البرمج العالية المستوى High level<br>بمعنى ان كل امر من لغة العالية المستوى يفسر الى عدة اوامر من لغة الاسيمبلى<br><br>لكى تتحول بعد ذلك الى لغة الاله<br>اذا هى الوسيط بين لغة عالية المستوى ولغة الاله | هى اللغه التى يفهمها الانسان ويستطيع التعامل معها ولكت لا تفهمها الاله<br><br>سهلة التعلم بالنسبه لنا<br><br>تستخدم فيها بعض الرموز الانجليزيه والتعبيرات الرياضيه<br><br>تحتاج الى وقت لكى تترجم الى لغة الاله<br><br>تتأثر البرامج التى برمجتها على اصدار معين من الويندوز عندما تعمل على اصدار اخر لكن تم تخطى هذه العقبه مع وجود Java virtual machine<br>.net frame work<br><br>امر واحد فيها يترجم الى عدة اوامر من الاسيمبلى |
| | | |

## Compilers

المترجمات هى نوع من البرامج والتى تكون النواه الاساسيه لاى لغة برمجه فى العالم حيث انها تمثل القاعده لها والتى تكون وسيلة او اداة التخطاب بين المبرمج وبين المعالج وبدونها لا يستطيع المبرمج ان يملى اوامره على جهاز الحاسوب وهى مصممه ايضا بواسطة لغات البرمجه لكى تفهم لغة عالية المستوى وتفهم ايضا لغة الاله

جدير بالذكر ان اشهر لغات البرمجه المنتجه للمترجمات هى لغة الباسكال والسى بلس بلس

ولها نوعان هما : -

Compilers – interpreter

لكن ما الفرق بينهما

## Compilers

تقوم هذه البرامج بانتاج نسخ قادره على العمل بشكل تنفيذى ومنفصل عن بيئة التطوير المتكامله التى انتاجها بواسطتها وتعمل على نظام التشغيل كما لو كان برنامج لديك

من المعروف ان بيئة التطوير المتكامله المقصود بها هى تلك النسخه البرمجيه الكامله التى تنتج بها برامجك مثلا كبرنماج

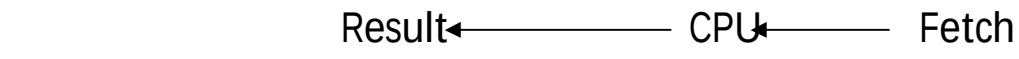
Visual studio 2008 – visual studio 6 – net beans

## Compiler operation sequence

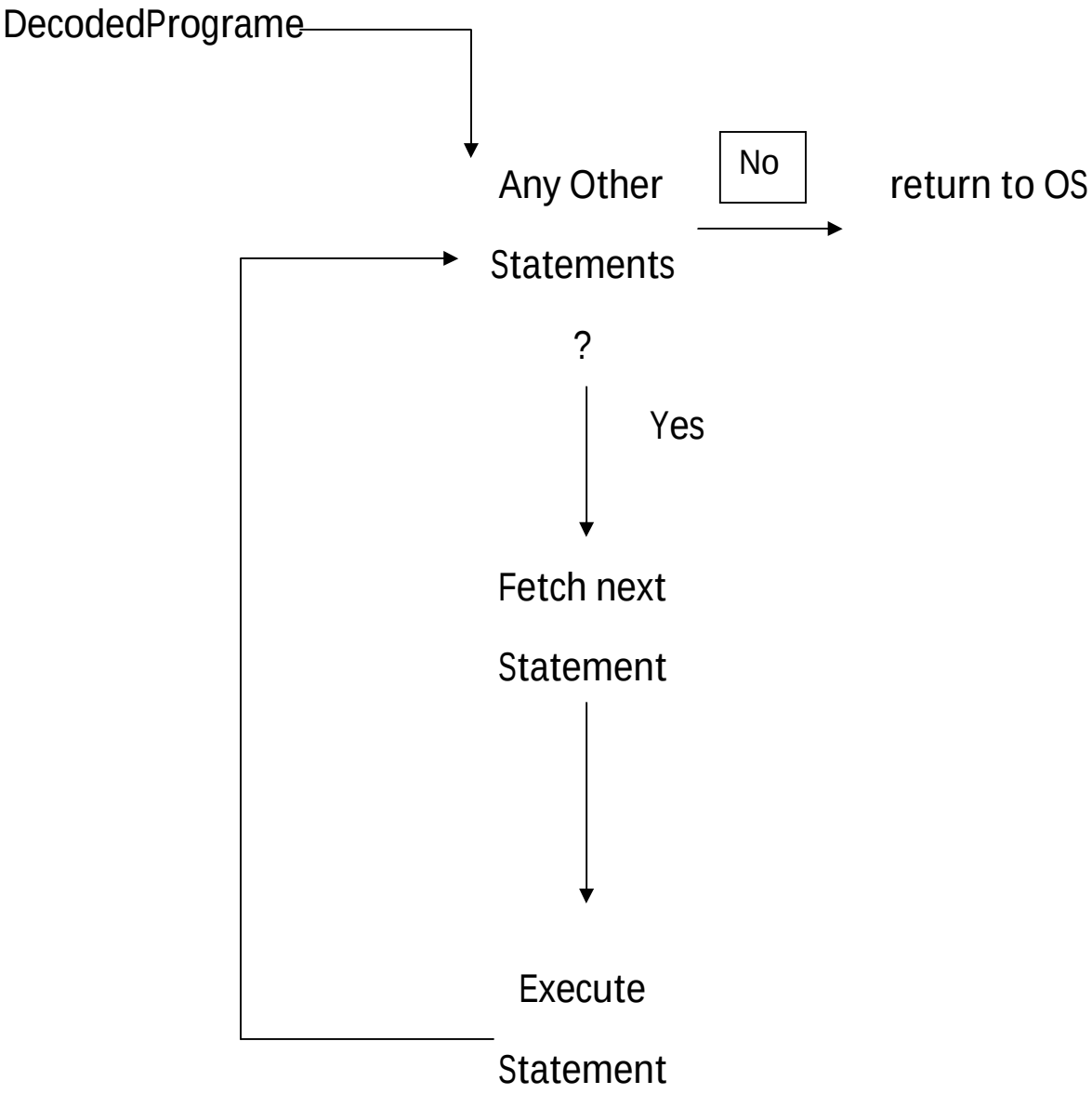
Using “Fetch – Execute” Cycle

وهى دورة التعرف والتنفيذ بمعنى انه يتعرف على الاوامر ثم يبداء التنفيذ

HLL Code ⟶ Compiler ⟶ organized sequence of commands




وعلى هذا فأن المسار التخطيطى او خريطة التدفق لهذه العمليه تكون كالتالى



من الواضح الان ان المترجم يقوم بالتعرف على الامر ثم تنفيذه وهكذا يدور فى حلقة من التعرف والتنفيذ الى ان ينتهى وبعدها يكون قادرا على اعطائك نسخته النهايئه والتى تعمل عليها كبرنماج منفصل عن بيئة التطوير

المترجمات تعمل عليها جميع لغات البرمجه التى تنتجها شركة مايكروسوفت بالاضافة الى لغات اخرى

## Interpreter

المفسر وهى يقوم بنفس عمل المترجم لكن الفرق ان المفسر لا يعطى نسخه تنفيذيه من البرنامج ويلزم وقت التنفيذ اتمام عملة التعرف على الامر

So it need decode while execute but interpreter dynamically for portability

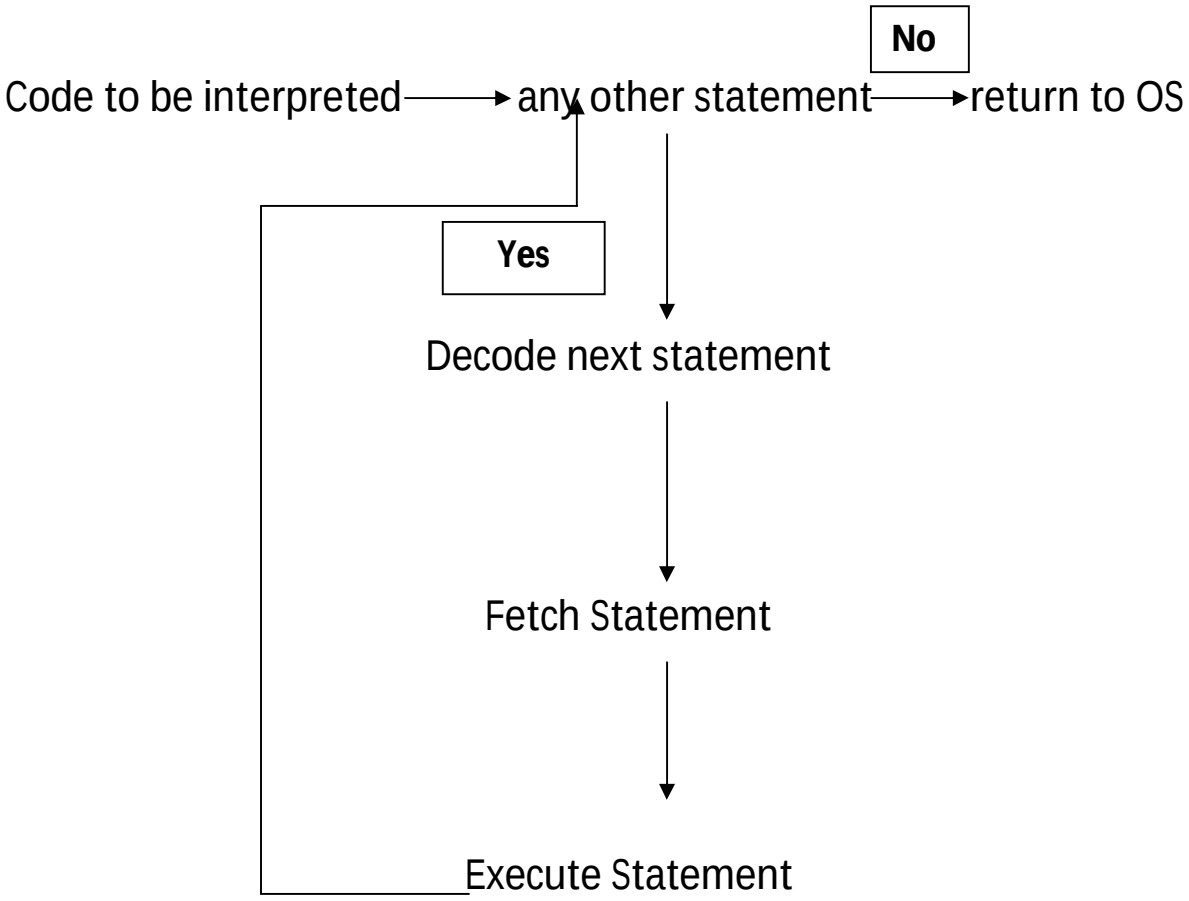
المفسر له قابلية حمل البرنامج ليعمل على اى نظام تشغيل وعلى اى متصفح وهذه ميزه هامه جدا ومثال على ذلك جميع البرامج التى تنتجها لغة الجافا وهذه الطريقه تستخدم

Decode – fetch – execute

Cycle

# Flow Chart

## Decode – fetch – execute



To understand mean of portability

لكى تفهم معنى كلمة ان تكون البرامج قابله للتشغيل على اى أله

.java ⟶ Java interpreter ⟶ .class as intermediate platform independent form

يكتب كود الجافا ثم يدخل على المفسر فينتج منه نسخه لاتعتمد على نوع الاله ويتم تشغيلها بواسطة

Java browser depend on java virtual machine

والتى لديها القدره على العمل على انظمة لينكس حتى لو كانت منتجه على نظام مايكروسوفت

**********
**********

# Development Cycle

## Software development life cycle SDLC

ان دورة حياة النظام تمر بمراحل متعدده وكل مرحله منها ذات اهميه كبيره لاتمام مشروعك بنجاح ودون وقوع اى اخطاء منطقيه ولها صور متعدده فمنها الطبقات المعتمده على بعضها ومنها لا يعتمد على بعضها البعض لكن سوف ناخذ نموذج بسيط وهو شلال المياه

### WATER FALL MODEL

Analysis

Design

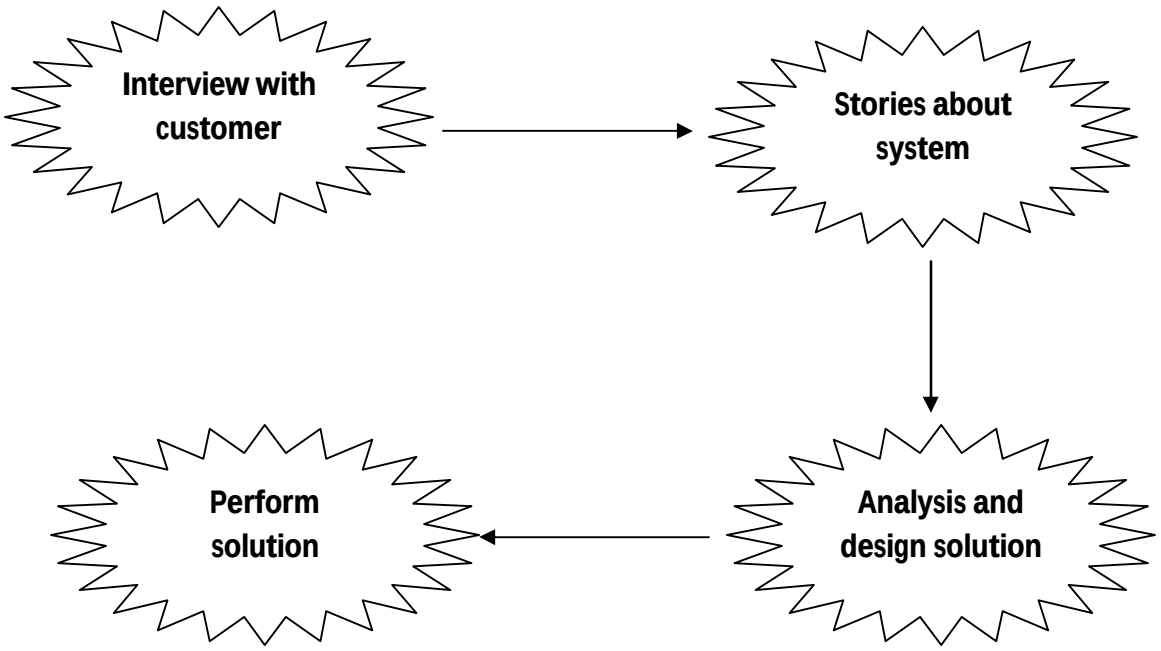
Coding

Testing

### Analysis:

تحليل متطلبات المشكله ودراستها جيدا لفهمها ووضع الخطوط العريضه فو صورة حكايات عن النظام ثم تلخيصها ووضعها بعد ذلك فى صورة تصميم والذى يتسطيع قاعدة المبرمجين من فهمه وترجمته الى كود فعلى وادخاله مرحلة التنفيذ

نستطيع عرض نموذج تخطيطى لما ذكرناه وتعتبر هذه مراحل دراسة النظام بشكل احترافى وممنهج




## Design:

تحديد كافة الادوات اللازمه لصنع النظام وممكن ان يكون لها شقين وهما الهاردوير والسوفت وير او شق واحد وهو السوفت وير لكن اى كان فهى تخضع لعدة مراحل

## Flow Chart or DFD

تتبع مسار النظام مداخل ومخارج النظام والعمليات التى تتم داخل النظام وهل للنظام امثر من دخل وهل سيتعامل مع شبكه اخرى من الانظمه وليس المقصود بالانظمه انها انظمة تشغيل

بل المقصود نظام الذى يتم صنعه لكى يدير شىء ما

## Algorithms

ما هى المتغيرات التى يتطلبها النظام لكى تقوم العمليات بالتفاعل مع بعضها واعطاء خرج فى النهايه فيجب تعريف جميع متغيرات النظام وانواع البيانات التى يتم ادخالها وكذلك تنظيم العمليات التى تتم عليها

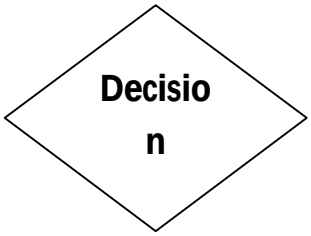
## Data Dictionary

كيف ينبغى ان يسير النظام بصوره تتوافق مع اداء مهامه المنوط بها دون وقوع خلل او اعطاء نتائج غير مرجوه

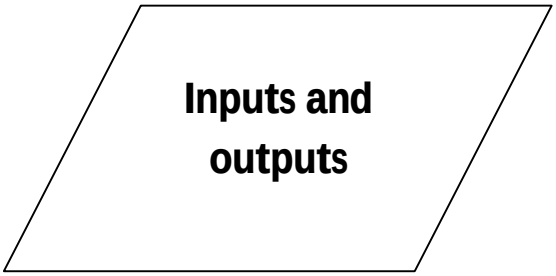
**********
**********

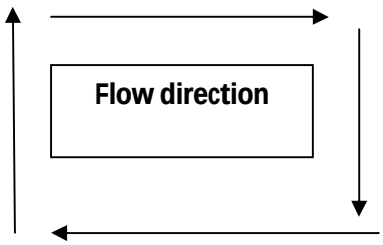
# Flow chart

وتستخدم فيها بعض الاشكال القياسيه وذات معنى وذلك لرسم مخطط البرنامج وكيفية حلها


Decisio
n

Process

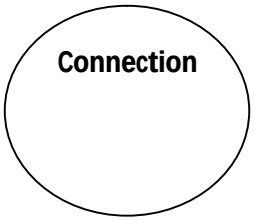

Inputs and
outputs


Flow direction

قد تكون تفاصيل البرنامج كبيره وليس من السهل حلها بواسطة مخطط واحد بل يمكن تقسيمها الى عدة مخططات مرتبطه مع بعضها ليسهل فهمها وهذا يحدث داخل العمليات او

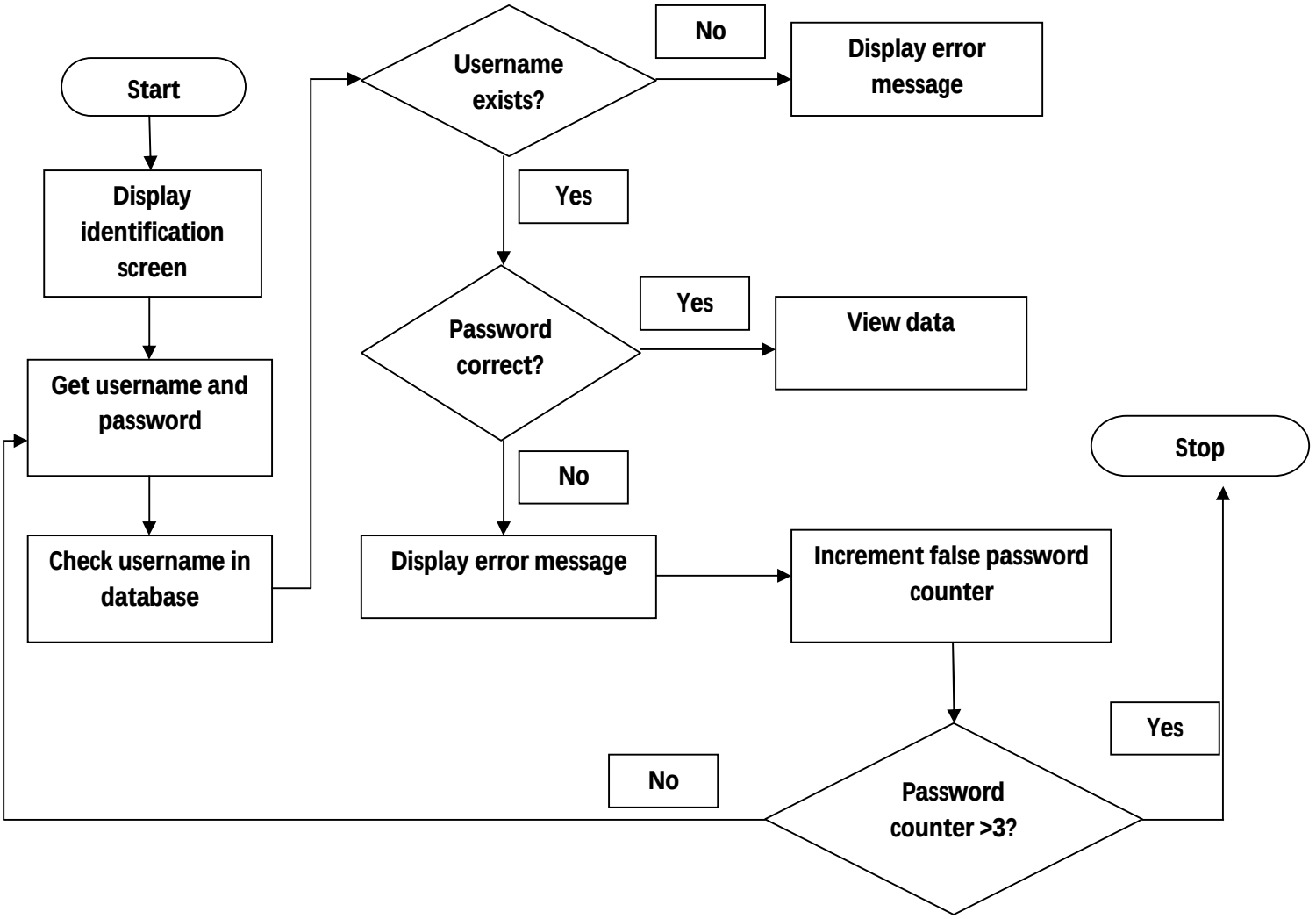
Process

ويتم الربط بينها بواسطة رمز




لدينا الان مثال برنامج يراد عمل تخطيط له وهذه البرنامج يمنع الدخول الا لمن لديه اسم المستخدم وكلمة السر

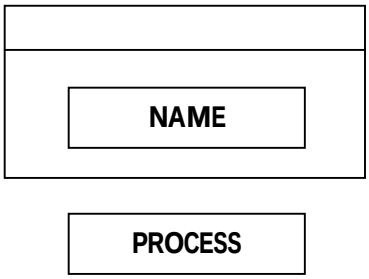
سوف يدخل الشخص اسم المستخدم وكلمة السر واذا كانت صحيحه سيتم الدخول على المعلومات وان لم تكن سيعطيه النظام فرصة ادخال بيانات حسابه 3 مرات متتاليه فقط



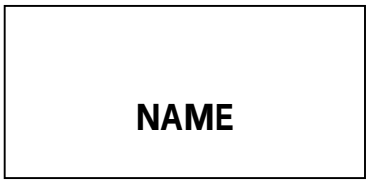
# Data Flow Diagram

مخطط تنفيذى للنظام والذى يتضح فيه بشكل كبير جميع المداخل والمخارج وجميع وظائف النظام الاساسيه وتعتبر جزء هام جدا من اجزاء التصميم

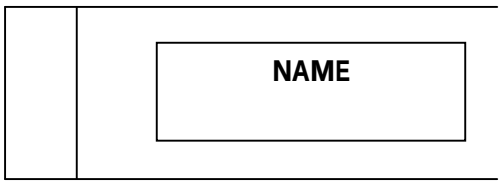
Basic shapes to draw DFDS



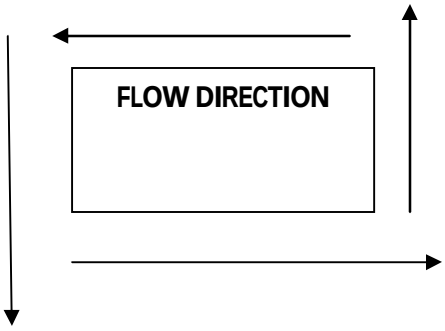

PROCESS




ENTITY




DATA STORE



هناك بعض البرامج المتخصصه فى تصميم خريطة تدفق البيانات

DRWAING TOOLS Visio and smart draw

لكن ما الذى يلزم لرسم مخطط تدفق بيانات صحيح اولا يجب ان تفرق بين

Entity – process

هى تعبر عن الاشخاص الذين سيتعاملون مع النظام ومن الممكن ان يكون اله Entity

وهى فى مثال السابق الشخص الذى سيقوم بادخال اسم المستخدم وكلمة السر

هى العمليه التى تتم على المدخلات وهى فى مثال السبق التحقق من صحة Process

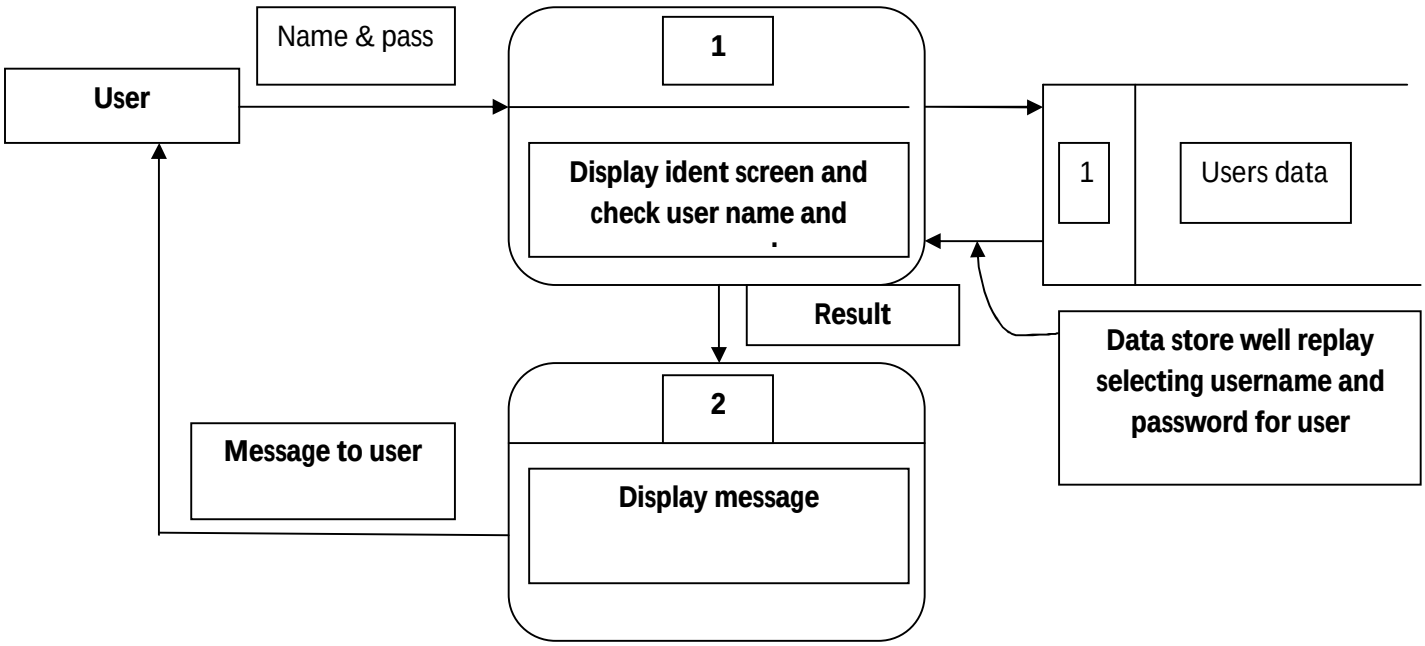
بيانات المستخدم

وهى قاعدة البيانات التى يبحث فيها النظام عن المعلومات المخزنه فيهData entry

وهى فى مثال السابق قاعدة بيانات المستخدمين

ويبقى عمليه اخيره وهى اظهار النتائج او

Display message



# Coding

وتعتبر اخر مرحله فى بناء النظام وليست النهائيه بمعنى انه بكتابة الكود نكون قد انتهينا فعليا من بناء نظام وبعد ذلك تحديد ما اذا كان يتم بناء واجه رسمويه للتعامل مع المستخدم ام انه يتعامل مع الالات مباشرتا ولا يحتاج الى واجه رسوميه ولابد بالطبع ان يكون المبرمج على درايه كامله بقواعد لغة البرمجه التى يستخدمها ويطور نفسه بشكل منتظم وسريع

# Testing and debugging

مرحله هامه جدا وهى اختبار وتصحيح اى خطاء منطقى او برمجى فى النظام ولكن ما هى انواعها

## Syntax error

وتعتبر الاسهل فى اكتشافها وتصحيحها وتعتبر هى الاقل خطرا والتى تظهر فى قواعد الكود ويمكن تداركها سريعا حيث ان بيئة التطوير الحديثه لكافة لغات البرمجه تعطى تنبيهات عديده بان هناك خطاء فى قواعد الكود ويجب عليك تصحيحها وبل و تحدد ايضا اين يقع الخطاء وسبب حدوثه

## Runtime error

تلك التى لا تظهر الا وقت التنفيذ وتعتبر خطيره نوعا ما وتسمى الاستثنائات وناتجه عن طريق العمليات الغير منطقيه كالقسمه على الصفر وغيره وهى بمثابة توقع الاخطاء التى يمكن ان يقع فيها المستخدم وعدم السماح له بالوقوع فيها حتى لا تؤثر على اداء البرنامج بل انها قد تسبب فى بعض الاحيان الى انهيار البرنامج

## Logical error

ويمثل النوع الاخطر لان كل شيء يبدو على ما يرام ولا يوجد اخطاء لغويه او منطقيه بل الخطاء سيكتشفه المستخدم عند استخدام البرنامج حيث ان البرنامج يعطى نتائج غير التى يرديها المستخدم

Required result not equal obtained result

ودائما تحدث نتيجة لعدم الترتيب المنطقى والسليم للعمليات النظام

- ينصح دائما بعدم عمل اى نظام او الدخول فى مرحلة الكود دون المرور بالمراحل التى تسبقه والا سيقع خلل رهيب فى اداء النظام

لا تستخدم لغة برمجه الا اذا كنت تعرف قواعدها جيدا لان هذه يضيع عليك الوقت والجهد

انتهى ويلى هذا الكتاب

Software Eng

هندسة البرمجيات

Sofyany – memorycode_84@yahoo.com