الأكاديمية العربية الدولية
Arab International Academy

اساسيات البرمجة

Programming Fundamentals

عبدالقادر العبدالله

كلية العلوم – تخصص البرمجة

المحاور

- مقدمة عن الحاسوب
- البيانات وأنواع البيانات
- المتغيرات واسناد القيم
- تخزين البيانات
- الخوارزميات والمخططات الانسيابية
- التعبيرات الحسابية والعلائقية والمنطقية

المخرجات المتوقعة من الدرس

- **فهم مفهوم الحاسوب ووظائفه الأساسية.**
- **معرفة الطالب لمفهوم البيانات والمعلومات** والفرق بينهما و يستطيع التمييز بين أنواع البيانات المختلفة.
- **الإلمام بمفهوم المتغيرات واستخداماتها في البرمجة** وفهم القواعد الرئيسية في تسمية المتغيرات.
- **التعرف على كيفية تخزين البيانات في الحاسوب** و الفرق بين الذاكرة المؤقتة والدائمة.
- **كتابة خوارزميات لحل مشكلة بسيطة** بخطوات مرتبة و رسم مخطط يمثل خوارزمية معينة.
- **فهم التعبيرات** الحسابية , التعبيرات العلائقية و التعبيرات المنطقية و تطبيقها.
- **ان يكون الطالب قادرا على التفكير المنطقي** وتحليل المشكلات البسيطة ورسم خوارزميات لحلها.

مقدمة عن الحاسوب

1 - الحاسوب (Computer) : هو جهاز إلكتروني يعمل على معالجة البيانات وتنفيذ العمليات الحسابية والمنطقية بسرعة فائقة ودقة عالية، بناء على مجموعة من التعليمات المبرمجة مسبقا :

- السرعة : يمكن للحاسوب معالجة ملايين العمليات الحسابية في الثانية الواحدة
- الدقة : ينفذ العمليات بدقة عالية دون أخطاء إذا كانت التعليمات المقدمة صحيحة
- التخزين : يستطيع تخزين كميات هائلة من البيانات والوصول إليها بسرعة
- الأتمتة : ينفذ المهام تلقائيا دون تدخل بشري بمجرد برمجته

مقدمة عن الحاسوب

2 - البت (Binary Digit-Bit) : هو أصغر وحدة تخزين في الحاسوب ويمثل إحدى حالتين فقط (0,1) تمثل وجود تيار كهربائي او عدمه و يستخدم الحاسوب النظام الثنائي لتمثيل و معالجة كل شيء من البيانات البسيطة مثل الأحرف إلى العمليات المعقدة و يتطلب ذلك نظاما محددا لتحويل البيانات إلى بتات منها :

a) الترميز الثنائي العشري (BCD - Binary Coded Decimal) :

- يعتمد على 4 بتات لتشفير الأرقام فقط
- كل رقم عشري (0 إلى 9) يمثل برمز مكون من 4 بتات

❖ هذا النظام لا يستطيع تمثيل الحروف والرموز، لذا يستخدم فقط للأرقام

Decimal Digit	BCD			
	8	4	2	1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

مقدمة عن الحاسوب

b) الترميز الموسع (EBCDIC) :

- يعتمد على 8 بتات لتمثيل الأحرف و الأرقام والرموز و يمكنه تمثيل 256 رمزا
- يسمح بتشفير عدد أكبر من الرموز مقارنة بـ BCD
- يستخدم غالبا في الحواسيب العملاقة

❖ لا يتوافق مع الأنظمة الحديثة ويتكون من تصميم معقد

EBCDIC											
Char	Binary Zone	Binary Digit	Hex	Char	Binary Zone	Binary Digit	Hex	Char	Binary Zone	Binary Digit	Hex
blank	0100	0000	40	u	1010	0100	A4	P	1101	0111	D7
a	1000	0001	81	v	1010	0101	A5	Q	1101	1000	D8
b	1000	0010	82	w	1010	0110	A6	R	1101	1001	D9
c	1000	0011	83	x	1010	0111	A7	S	1110	0010	E2
d	1000	0100	84	y	1010	1000	A8	T	1110	0011	E3
e	1000	0101	85	z	1010	1001	A9	U	1110	0100	E4
f	1000	0110	86	A	1100	0001	C1	V	1110	0101	E5
g	1000	0111	87	B	1100	0010	C2	W	1110	0110	E6
h	1000	1000	88	C	1100	0011	C3	X	1110	0111	E7
i	1000	1001	89	D	1100	0100	C4	Y	1110	1000	E8
j	1001	0001	91	E	1100	0101	C5	Z	1110	1001	E9
k	1001	0010	92	F	1100	0110	C6	0	1111	0000	F0
l	1001	0011	93	G	1100	0111	C7	1	1111	0001	F1
m	1001	0100	94	H	1100	1000	C8	2	1111	0010	F2
n	1001	0101	95	I	1100	1001	C9	3	1111	0011	F3
o	1001	0110	96	J	1101	0001	D1	4	1111	0100	F4
p	1001	0111	97	K	1101	0010	D2	5	1111	0101	F5
q	1001	1000	98	L	1101	0011	D3	6	1111	0110	F6
r	1001	1001	99	M	1101	0100	D4	7	1111	0111	F7
s	1010	0010	A2	N	1101	0101	D5	8	1111	1000	F8
t	1010	0011	A3	O	1101	0110	D6	9	1111	1001	F9

مقدمة عن الحاسوب

c) الترميز الأمريكي القياسي (ASCII) :

- الأكثر شيوعا على مستوى العالم وخاصة في الحواسيب الشخصية
- يعتمد على 7 بتات في نسخته الأصلية مما يسمح بتمثيل 128 رمزا
- اضيف بت اضافي ليصبح النظام مكونا من 8 بتات ويمثل 256 رمزا
- يشمل الأحرف, الأرقام و الرموز

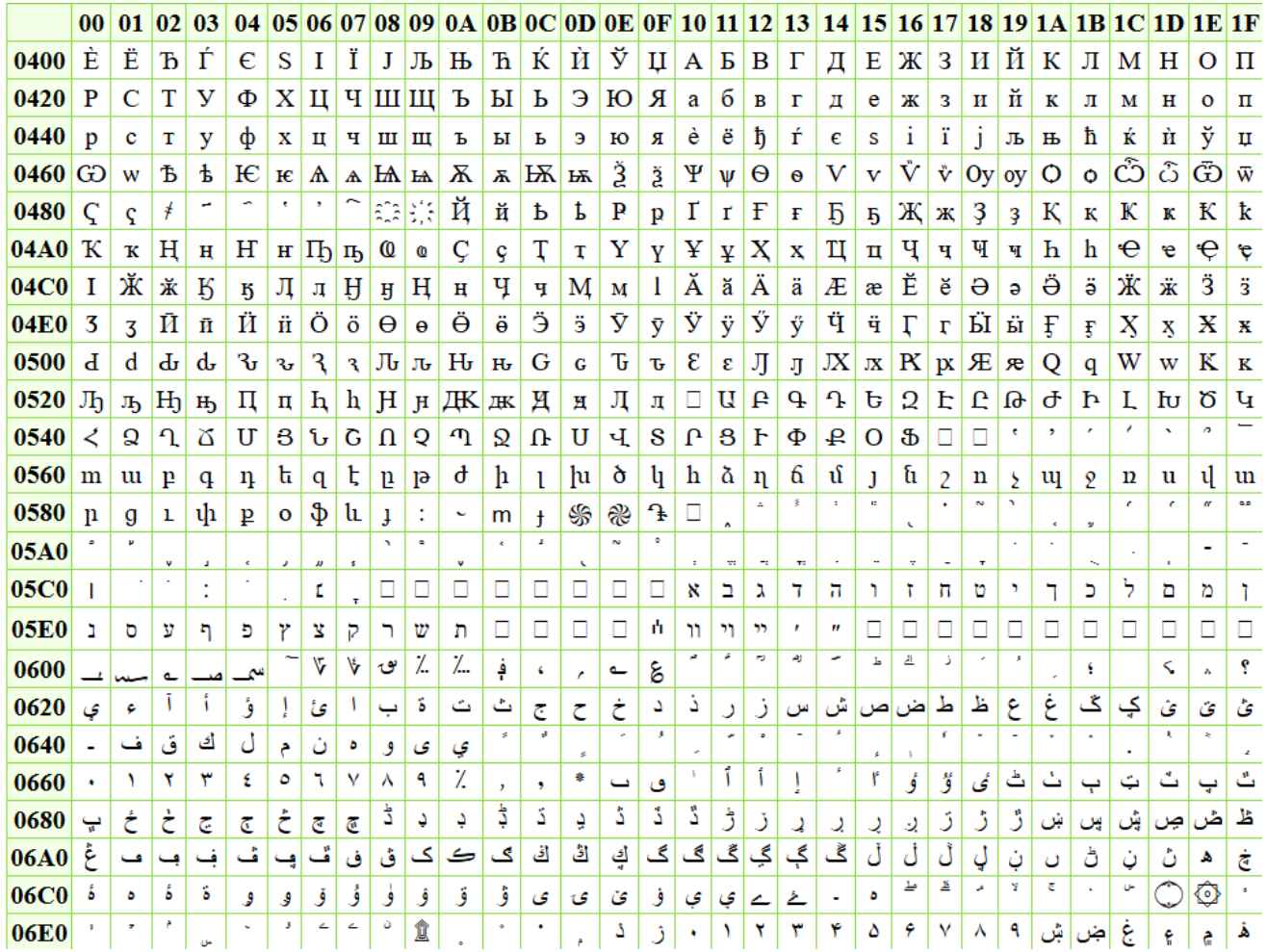
ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

مقدمة عن الحاسوب

d) الترميز الشامل (Unicode) :

- يستخدم لتمثيل النصوص في جميع اللغات حول العالم
- يعتمد على مجموعة بتات متغيرة (مثل 16 أو 32 بت) لتمثيل ملايين الرموز
- مناسب للغات التي تحتوي على عدد كبير من الأحرف مثل العربية، الصينية، والهندية



مقدمة عن الحاسوب

من حيث	Unicode	ASCII	EBCDIC	BCD
الاستخدامات	تمثيل النصوص العالمية (جميع اللغات والرموز)	النصوص الإنجليزية وبعض الرموز العامة	التطبيقات التجارية الكبيرة	تمثيل الأرقام العشرية فقط
دعم اللغات	يدعم جميع اللغات، مثل العربية، الصينية	يدعم النصوص الإنجليزية فقط	يدعم النصوص الإنجليزية وبعض الرموز	لا يدعم اللغات، يركز فقط على الأرقام
ترتيب الرموز	منطقي ومصمم لدعم اللغات المختلفة بترتيب موحد	منطقي وبسيط (الرموز النصية مرتبة بشكل تسلسلي)	معقد وغير بديهي مقارنة بـ ASCII	غير مناسب للنصوص أو الرموز
الاستخدام الحالي	يستخدم في جميع الأنظمة الحديثة	يستخدم في بعض الأنظمة التقليدية و القديمة	نادر الاستخدام خارج انظمة IBM	نادر الاستخدام بسبب محدوديته
المرونة	مرن جدا وقابل للتوسيع	محدود في عدد الرموز المدعومة	محدود ويقتصر على التطبيقات التجارية القديمة	محدود للغاية بسبب دعم الأرقام فقط

مقدمة عن الحاسوب

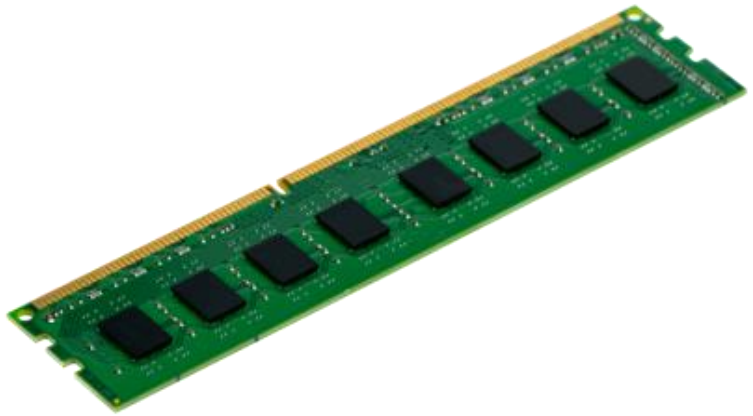
3 - مكونات الحاسوب (Computer Components) : الحاسوب يتكون من عنصرين رئيسيين:

a) الأجهزة (Hardware) :

- وحدة المعالجة المركزية (CPU) : هي عقل الحاسوب حيث يتم تنفيذ العمليات الحسابية والمنطقية ومعالجة البيانات تتألف من :
  - المعالج (Processor) : ينفذ التعليمات ويقوم بمعالجة البيانات
  - المسجلات (Registers) : وحدات تخزين صغيرة داخل المعالج تستخدم لتخزين البيانات مؤقتا أثناء المعالجة
  - وحدة التحكم (Control Unit): تدير تدفق البيانات بين أجزاء الحاسوب
  - الوحدة الحسابية والمنطقية (ALU): تنفذ العمليات الحسابية والمنطقية

مقدمة عن الحاسوب

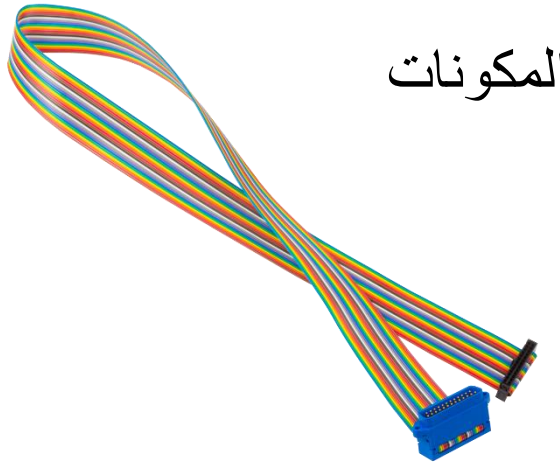
- ذاكرة الوصول العشوائي (RAM-Random Access Memory) : ذاكرة مؤقتة تستخدم لتخزين البيانات والتعليمات التي يحتاجها ال CPU اثناء التشغيل :
  - تعتبر ذاكرة مؤقتة أي تفقد محتوياتها عند انقطاع الطاقة
  - تستخدم لتحميل البرامج والبيانات التي يتم تشغيلها حاليا
  - تقاس سعة RAM بوحدات البايت (Bytes)
  - تتيح RAM تشغيل تطبيقات متعددة في الوقت نفسه



❖ كلما زادت كمية RAM زادت قدرة الحاسوب على تشغيل التطبيقات الثقيلة بكفاءة.

مقدمة عن الحاسوب

- النواقل (Bus) : هي خطوط اتصال تستخدم لنقل البيانات والتعليمات بين المكونات الرئيسية للحاسوب مثل وحدة المعالجة المركزية (CPU) وذاكرة الوصول العشوائي (RAM) وباقي الأجهزة الطرفية ولها عدة انواع منها :
  - ناقل البيانات : نقل البيانات بين المكونات المختلفة
  - ناقل العناوين : يستخدم لنقل عناوين المواقع في الذاكرة التي تحتاج CPU للوصول إليها
  - ناقل التحكم : ينقل إشارات التحكم والإشارات الزمنية بين المكونات



مقدمة عن الحاسوب

- أجهزة الإدخال والإخراج (I/O Devices) : أجهزة الإدخال والإخراج هي الوسيلة التي من خلالها يتفاعل المستخدم مع الحاسوب و توفر هذه الأجهزة القدرة على إدخال البيانات إلى الحاسوب وإخراج المعلومات منه
- الساعة النظامية (System Clock): الساعة النظامية هي إحدى المكونات الأساسية لوحدة المعالجة المركزية (CPU) تصدر نبضات كهربائية منتظمة تستخدم لتنظيم توقيت العمليات داخل الحاسوب و تعمل على تنسيق العمليات و تحديد السرعة التي يمكن لوحدة المعالجة المركزية تنفيذها في الثانية.




مقدمة عن الحاسوب

b) البرمجيات (Software) : تتكون من مجموعة من البرامج التي تعمل على تشغيل النظام ومعالجة البيانات وتقوم بتوجيه الأجهزة (Hardware) لأداء العمليات المختلفة وتقسم الى ثلاث انواع رئيسية :

- البرامج الثابتة (Firmware) : تعتبر البرمجيات الأساسية التي تقوم بتشغيل الأجهزة عند بدء التشغيل تخزن عادة في ذاكرة ROM ذاكرة القراءة فقط و تقوم البرامج الثابتة بتخزين إعدادات تكوين النظام
- نظام التشغيل (Operating System): البرنامج الرئيسي الذي يدير جميع مكونات الحاسوب و يوفر واجهة للمستخدم للتفاعل مع الحاسوب .

- برامج التطبيقات (Application Software): البرامج التي تستخدم لتنفيذ مهام محددة تلبي احتياجات المستخدم

مقدمة عن الحاسوب

- إدارة المعالج (Processor Management) : تنظم هذه الوحدة استخدام وحدة المعالجة المركزية لضمان تنفيذ جميع العمليات و تقوم بإدارة قائمة الانتظار للعمليات و تحدد فترة زمنية ثابتة لكل عملية مع مراقب حالة كل عملية أثناء تنفيذها لضمان اكتمال التنفيذ.
- إدارة الذاكرة (Memory Management): تعتمد الحواسيب الحديثة على معمارية البرنامج المخزن (Stored Program Architecture)حيث يجب أن يكون البرنامج قيد التنفيذ موجودا في الذاكرة الأساسية (RAM) ومن وظائف وحدة إدارة الذاكرة :
  - مراقبة حالة الRAM و التأكد من أن كل جزء من الذاكرة يعمل بشكل صحيح
  - تخصيص مساحة ذاكرة للبرامج بناء على احتياجاتها وتوافر المساحة
  - ضمان عدم انتهاك المساحة المخصصة لكل برنامج بواسطة برامج أخرى

مقدمة عن الحاسوب

- إدارة الأجهزة (Device Management) : هي جزء من نظام التشغيل يختص بالتحكم في جميع الأجهزة المتصلة بالحاسوب :
  - يحافظ على قائمة بجميع الأجهزة المتصلة بالحاسوب و يراقب حالة تشغيلها وأدائها
  - يتعامل مع الطلبات الصادرة من العمليات للوصول إلى الأجهزة
- إدارة الشبكة (Network Management): هذه الوحدة تسهل الاتصال بين الحاسوب والشبكة وتدير إرسال واستقبال البيانات :
  - تفحص كل حزمة بيانات مستقبلة للتأكد من ان الحزمة موجهة إلى الحاسوب او يتم تجاهلها
  - تقسيم البيانات المراد إرسالها إلى حزم صغيرة ضمان توافق الحزم مع بروتوكول الشبكة المستخدم

مقدمة عن الحاسوب

من حيث	إدارة الذاكرة	إدارة الشبكة	إدارة الأجهزة	إدارة المعالج
المهمة الرئيسية	تخصيص الذاكرة للبرامج وضمان عدم انتهاك البرامج للمساحة	إرسال واستقبال البيانات عبر الشبكة مع ضمان الأمان والكفاءة	إدارة التفاعل بين الحاسوب والأجهزة الطرفية المتصلة	تخصيص وقت المعالج للعمليات وضمان استكمال تنفيذها بكفاءة
الموارد المدارة	الذاكرة والذاكرة الافتراضية	بطاقات الشبكة والبروتوكولات	الأجهزة مثل الطابعات،الأقراص الصلبة	وحدة المعالجة المركزية وسجل العمليات
التكامل مع النظام	متكاملة مع إدارة المعالج لتخزين التعليمات والبيانات	متكاملة مع إدارة الأجهزة لتسهيل الاتصالات	تعتمد على إدارة الذاكرة لتخصيص الموارد المؤقتة	تعتمد على الذاكرة لتخزين التعليمات وتنفيذها
الأهمية	تسهيل تشغيل البرامج متعددة المهام وتقليل التجزئة	تحسين التواصل مع الشبكة وضمان سرعة نقل البيانات	تمكين الحاسوب من استخدام الأجهزة بكفاءة وسهولة	تحسين كفاءة الأداء من خلال توزيع وقت المعالجة بعدالة

البيانات وأنواع البيانات

1 - البيانات (Data) : مجموعة من القيم الأولية التي تمثل معلومات حول كائنات أو كيانات أو أحداث و تعتبر البيانات الأساس الذي تبنى عليه المعلومات وهي لا تحمل معنى واضحا بذاتها دون معالجة أو تنظيم وتصنف الى :

a) حسب المصدر :

- بيانات أولية (Primary Data): تجمع مباشرة من أجهزة أو برامج الحاسوب [البيانات المدخلة عبر لوحة المفاتيح]
- بيانات ثانوية (Secondary Data): تستخرج من قواعد بيانات أو مصادر موجودة مسبقا[السجلات المحفوظة في نظام إدارة الملفات]

البيانات وأنواع البيانات

b) حسب طبيعتها :

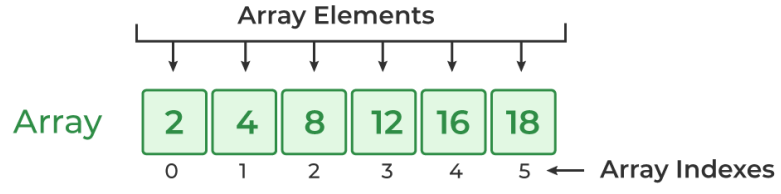
- بيانات كمية (Quantitative Data): تعبر عن قياسات أو كميات[عدد العمليات المنفذة في المعالج]
- بيانات نوعية (Qualitative Data): بيانات وصفية تعبر عن خصائص أو سمات[اسم المستخدم]

- ومن خصائص البيانات :
  - قابلة للتخزين : تخزن البيانات في وسائط التخزين المختلفة مثل الأقراص الصلبة
  - قابلة للمعالجة : يمكن للحاسوب معالجة البيانات لتحويلها إلى مخرجات مفيدة
  - قابلة للتنظيم : تنظم البيانات في هياكل مثل الجداول (Tables) أو المصفوفات (Arrays)

البيانات وأنواع البيانات

2 - أنواع البيانات الخاصة (Special Data Types) : أنواع من البيانات التي تستخدم لأغراض خاصة في البرامج وخاصة في تطوير برامج النظام :

- المصفوفات (Arrays) : المصفوفات هي جداول بيانات تستخدم لتخزين مجموعة من القيم المتشابهة تستخدم بشكل كبير في حل المشكلات الرياضية ومن انواعها :
  - مصفوفة أحادية البعد (Single Dimensional Array)
  - مصفوفة ثنائية الأبعاد(Two-Dimensional Array)



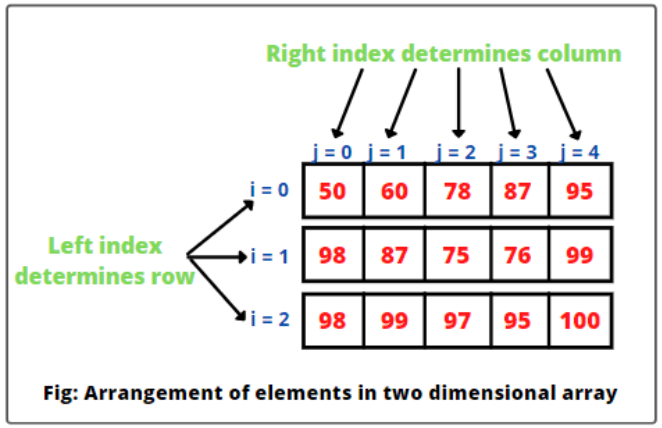
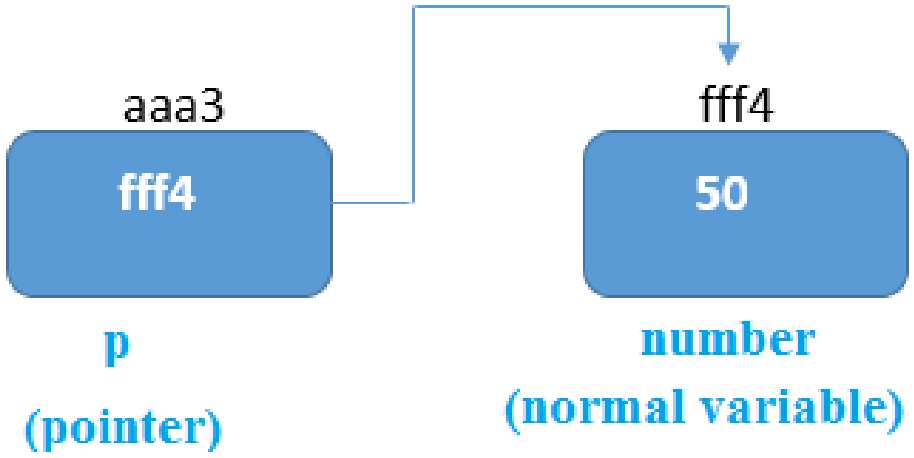


Fig: Arrangement of elements in two dimensional array

❖ المصفوفات تخزن في قطعة متجاورة من ذاكرة RAM ويعتمد حجم المصفوفة على نوع البيانات وعدد العناصر.

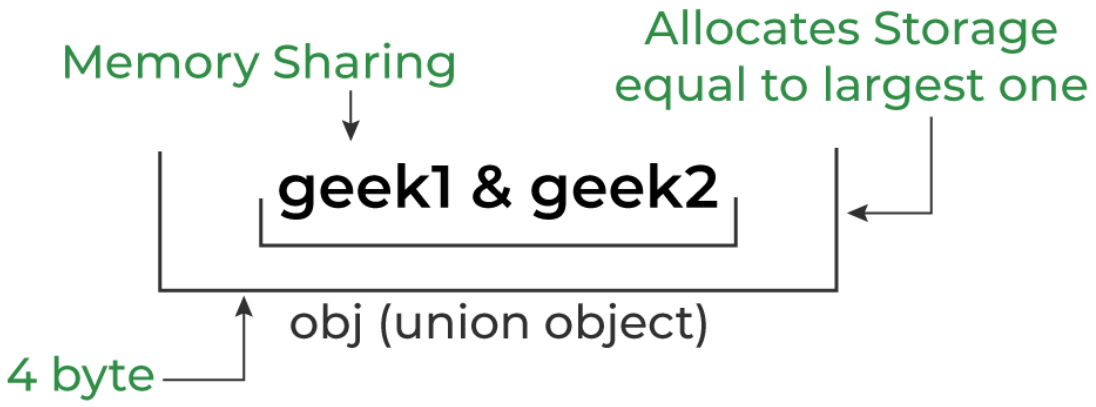
البيانات وأنواع البيانات

- المؤشرات (Pointers) : المؤشرات هي نوع خاص من البيانات يمكنه حفظ عنوان موقع ذاكرة في ال(RAM) و يتمثل حجم المؤشر بعدد البايتات المطلوبة لتخزين أكبر عنوان في الذاكرة :
  - تستخدم لإدارة تخصيص وإلغاء تخصيص الذاكرة
  - تتيح الوصول المباشر إلى أي موقع في الذاكرة
  - تستخدم للتعامل مع المصفوفات



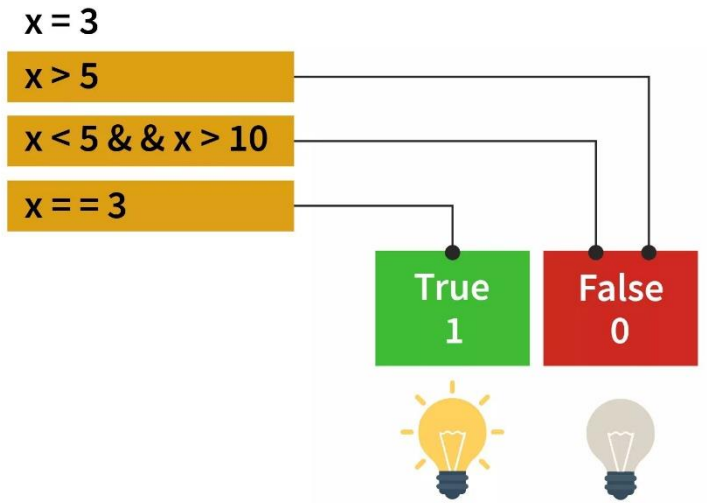
البيانات وأنواع البيانات

- الاتحاد (Union) : نوع بيانات خاص في لغات البرمجة يسمح باستخدام نفس مساحة الذاكرة لعدة متغيرات بأنواع بيانات مختلفة ويمكن أن يخزن بيانات رقمية وحرفية معا:
  - مفيد في التطوير الفوري (Real-Time Development) حيث تكون الذاكرة محدودة



البيانات وأنواع البيانات

- البيانات المنطقية (Boolean) : نوع بيانات يحتوي على حالتين فقط 1 صحيح و 0 خطأ ويستخدم في :
  - الإشارات (Flags): إذا تم تغيير مستند يتم ضبط العلم إلى True وإذا لم يتم تغييره يبقى False
  - الإشارات المنطقية (Logical signals) :التحقق من وجود سجلات إضافية في جدول قاعدة بيانات
  - في أنواع البرمجيات (Software): مثل التحقق من الحالات و إدارة الحالات الخاصة



البيانات وأنواع البيانات

من حيث	Arrays	Pointers	Union	Boolean
الوظيفة	تخزين مجموعات بيانات متجانسة	تخزين عناوين الذاكرة RAM	مشاركة نفس مساحة الذاكرة لعدة أنواع بيانات	تخزين حالتين منطقيتين فقط
التخزين	قطعة متجاورة من RAM	حجم ثابت يعتمد على الذاكرة	مساحة مشتركة لعدة متغيرات	عادة بت واحد (ولكن تخزن كـ بايت)
الاستخدام	العمليات الحسابية والمصفوفات	الوصول المباشر للذاكرة	تحسين استخدام الذاكرة	تتبع الحالات أو الإشارات
الأمان	آمن	قد يسبب مشكلات أمنية إذا أسيء استخدامه	يعتمد على المبرمج	آمن تماما
المجالات	الرياضيات، البرمجيات التجارية	برمجيات النظام، إدارة الذاكرة	البرمجيات الفورية حيث تكون الذاكرة محدودة.	تتبع الأحداث أو الحالات

البيانات وأنواع البيانات

3 - تصنيف البيانات (Data Classes) : تصنف البيانات الى نوعين اساسيين :

- البيانات المحلية (Local Data) : هي البيانات التي تعلن داخل برنامج فرعي (Subprogram)، وتكون متاحة فقط للاستخدام داخل ذلك البرنامج الفرعي و من خصائصها :
  - النطاق (Scope) : تكون مرئية فقط للبرنامج الفرعي الذي تم الإعلان عنها فيه
  - العمر (Lifetime): يتم تحريرها وإرجاعها إلى نظام التشغيل بعد انتهاء تنفيذ البرنامج الفرعي
  - العزل (Insulation): غير متاحة للبرامج الفرعية الأخرى أو البرنامج الرئيسي
  - الإعداد الافتراضي (Default setting): المتغيرات المعلنة داخل البرامج الفرعية تعتبر محلية بشكل افتراضي

البيانات وأنواع البيانات

- البيانات العامة (Global Data) : هي البيانات التي تعلن في البرنامج الرئيسي أو يتم تحديدها كـ "عامة" في أحد البرامج الفرعية لتكون متاحة لجميع البرامج الفرعية الأخرى و من خصائصها :
  - النطاق (Scope) : متاحة لجميع البرامج الفرعية التي يتم استدعاؤها بواسطة البرنامج الرئيسي
  - العمر (Lifetime): تبقى مشغولة في الذاكرة طوال فترة تشغيل البرنامج الرئيسي وكل برامجه الفرعية
  - الإعداد الافتراضي (Default setting): المتغيرات المعلنة في البرنامج الرئيسي تعتبر عامة بشكل افتراضي

البيانات وأنواع البيانات

من حيث	البيانات المحلية	البيانات العامة
الاستهلاك	تستهلك ذاكرة لفترة قصيرة فقط	تشغل ذاكرة لفترة طويلة مما يزيد الحمل على RAM
الأمان	معزولة وآمنة من التداخل بين البرامج الفرعية	عرضة للتعديل من قبل أي برنامج فرعي، مما قد يسبب أخطاء
الاستخدام	لحالات الاستخدام المحدودة داخل برنامج فرعي واحد	للقيم التي تحتاجها عدة برامج فرعية أو البرنامج الرئيسي

المتغيرات واسناد القيم

1 - المتغيرات (Variables) : هو مكان يتم حجزه في الذاكرة لتخزين بيانات اثناء تشغيل البرنامج و يتم ربط المتغير بنوع معين من البيانات ويستخدم كمرجع لموقع مخصص في الذاكرة (RAM) ومن خصائصه :

- التسمية (Naming) : اسم المتغير هو معرف فريد يستخدم للإشارة إلى البيانات و تختلف قواعد التسمية بين لغات البرمجة
- الارتباط بنوع البيانات (Data Type) : كل متغير يتم ربطه بنوع بيانات محدد و يحدد نوع البيانات حجم الذاكرة المطلوب وطريقة معالجة البيانات
- مرجعية الذاكرة (Memory Reference): عند استخدام المتغير في البرنامج، يشير إلى الموقع المخصص له في الذاكرة

المتغيرات واسناد القيم

2 - أنواع المتغيرات في البرمجة (Variable Types) :

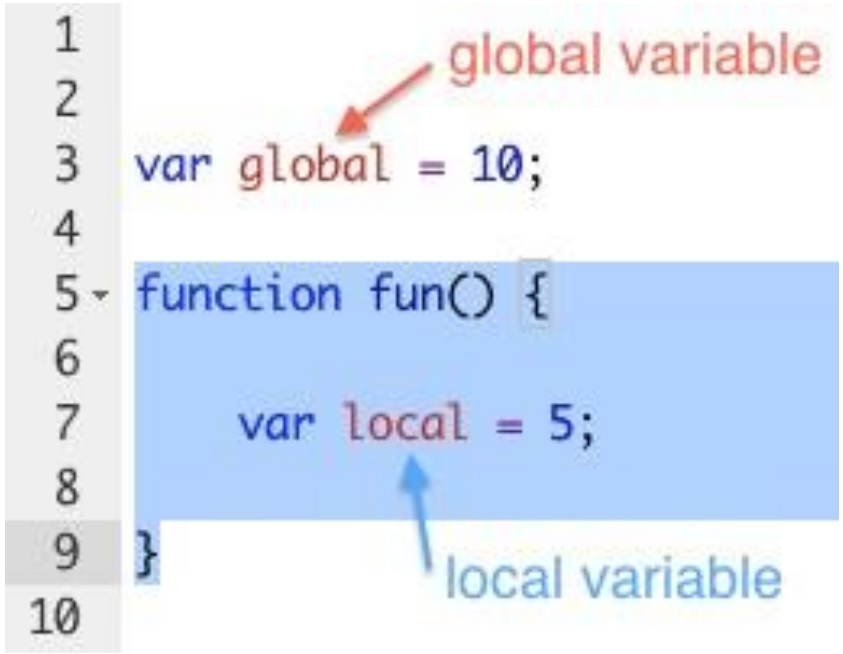
- **المتغيرات المحلية (Local Variables) :** المتغيرات التي يتم الإعلان عنها داخل دالة أو كتلة برمجية ويكون نطاقها محصورا داخل تلك الدالة وتخزين القيم المؤقتة التي لا يحتاج البرنامج الاحتفاظ بها بعد انتهاء تنفيذ الدالة :
  - يمكن الوصول إليها فقط داخل الدالة أو الكتلة التي تم تعريفها فيها
  - يتم تحرير ذاكرتها عند انتهاء تنفيذ الدالة
  - يتم تهيئتها عادة بقيمة ابتدائية عند الإعلان عنها

```
3     public static void main(String[] args)
4     {
5         double radius; // Declare radius
6         double area; // Declare area
7
8         // Assign a radius
9         radius = 20; // New value is radius
10
```

Local variables

المتغيرات واسناد القيم

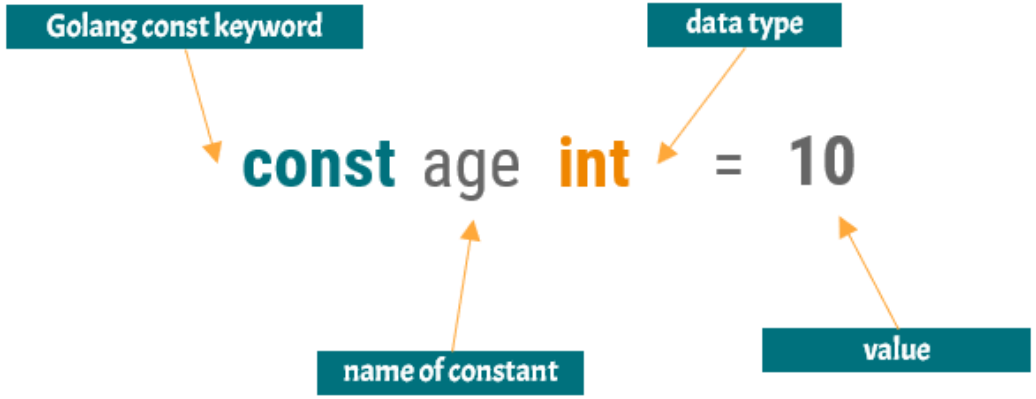
- **المتغيرات العامة (Global Variables) :** المتغيرات التي يتم تعريفها خارج جميع الدوال أو الكتل البرمجية، وتكون متاحة لجميع أجزاء البرنامج و تستخدم لتخزين القيم التي تحتاج إلى مشاركتها بين أجزاء متعددة من البرنامج :



- يمكن الوصول إليها وتعديلها من أي دالة في البرنامج
- تظل محتفظة بقيمتها طوال فترة تشغيل البرنامج
- قد تؤدي إلى أخطاء إذا تم تعديلها من قبل أكثر من دالة دون ضبط

المتغيرات واسناد القيم

- **المتغيرات الثابتة (Constant Variables)** :المتغيرات التي لا يمكن تغيير قيمتها بعد تهيئتها وتستخدم في تخزين القيم الثابتة مثل الثوابت الرياضية :
  - تعرف باستخدام الكلمة المفتاحية Const في العديد من لغات البرمجة
  - تستخدم لتخزين القيم التي لا تتغير أثناء تشغيل البرنامج



المتغيرات واسناد القيم

- **المتغيرات الديناميكية (Dynamic Variables) :** المتغيرات التي يتم تخصيص ذاكرتها أثناء تشغيل البرنامج باستخدام إدارة الذاكرة الديناميكية و تستخدم عندما لا نعلم حجم البيانات المطلوب أثناء كتابة الكود :
  - يتم تخصيصها باستخدام الدوال أو الكلمات المفتاحية مثل new و malloc
  - تحتاج إلى تحرير الذاكرة يدويا باستخدام delete او free

```
int main() {
int* ptr = new int;
*ptr = 42;
delete ptr;
    return 0;}
```

```
int main() {
int* ptr = (int*)malloc(sizeof(int));
*ptr = 42;
free(ptr);
  return 0;}
```

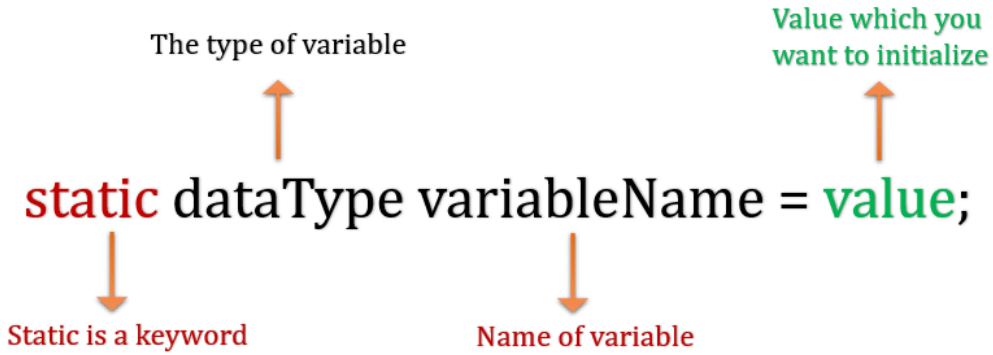
المتغيرات واسناد القيم

- **المتغيرات المؤقتة (Temporary Variables) :** تُستخدم لفترة قصيرة أثناء العمليات المؤقتة مثل الحسابات أو التبديل بين القيم:
  - تستخدم لتخزين القيم الوسيطة أثناء العمليات
  - عادة ما تكون متغيرات محلية
  - تستخدم مرة واحدة أو لفترة زمنية قصيرة

```
void swap(int &a, int &b) {
    int temp = a;
    a = b;
    b = temp;
}
```

المتغيرات واسناد القيم

- **المتغيرات الساكنة (Static Variables) :** المتغيرات التي تحتفظ بقيمتها حتى بعد انتهاء تنفيذ الدالة التي أُعلنت فيها:
  - تستخدم للحفاظ على حالة أو قيمة معينة عبر استدعاءات متعددة للدالة
  - يتم تعريفها باستخدام الكلمة المفتاحية static
  - تخزن في الذاكرة الثابتة (Static Memory) وليس في (Stack)



المتغيرات واسناد القيم

النوع	النطاق	العمر	الاستخدام الرئيسي
المتغيرات المحلية	داخل الدالة فقط	ينتهي بانتهاء تنفيذ الدالة	تخزين البيانات المؤقتة
المتغيرات العامة	جميع أجزاء البرنامج	يستمر طوال مدة تشغيل البرنامج	مشاركة القيم بين الدوال
المتغيرات الثابتة	يعتمد على مكان تعريفها	لا يمكن تغيير قيمتها	تخزين القيم الثابتة
المتغيرات الديناميكية	حسب تخصيص الذاكرة	يتم تحريرها يدويا	تخزين البيانات الديناميكية
المتغيرات المؤقتة	داخل الدالة فقط	قصير جدا أثناء العملية	العمليات الوسيطة
المتغيرات الساكنة	داخل الدالة، لكن قيمتها محفوظة	تستمر طوال مدة تشغيل البرنامج	الحفاظ على حالة محددة بين استدعاءات الدوال

المتغيرات واسناد القيم

2 - أنواع البيانات (Data Types) : أنواع البيانات هي الأساس الذي يعتمد عليه أي برنامج لتخزين ومعالجة المعلومات.

يتم تحديد نوع البيانات لتحديد حجم الذاكرة المطلوبة للتخزين و طريقة تمثيل ومعالجة البيانات :

a) الأنواع الأساسية (Primitive Data Types) :

- البيانات العددية الصحيحة (int-Integer)

- يستخدم لتخزين الأعداد الصحيحة بدون فواصل عشرية

- يتألف من 16 بت او 32 بت او 64 بت (Short Integer , Integer , Long Integer)

- اذا لم يتم تعيين قيمة تعيين 0 بشكل افتراضي

```
int main()
{
    short int miles = 20;
    int checking = 24;
    long diameter = 34000;
```

short integer literal
integer literal
long integer literal

المتغيرات واسناد القيم

- البيانات العشرية (Float) :
  - دقة عادية
  - يتألف من 4 بايت = 32 بت
  - اذا لم يتم تعيين قيمة تعيين f0.0 بشكل افتراضي
- البيانات العشرية (Double) :
  - دقة اعلى
  - يتألف من 8 بايت = 64 بت
  - اذا لم يتم تعيين قيمة تعيين d0.0 بشكل افتراضي

المتغيرات واسناد القيم

- البيانات الحرفية (Character) :
  - تستخدم لتخزين حرف واحد فقط
  - يتألف من 1 بايت = 8 بت
  - يمثل الأحرف باستخدام جداول ترميز مثل ASCIIأو Unicode

```
char letter = 0;
cout << "The size of char type is: " << sizeof(letter) << " byte." << endl;

letter = 'A';
cout << "letter = " << letter << endl;
letter = 'B';
cout << "letter = " << letter << endl;
```

character literals

- البيانات النصية (String) :
  - تستخدم لتخزين النصوص
  - يتم تمثيل النصوص كسلسلة متتابعة من الأحرف في الذاكرة، حيث يتم تخصيص موقع لكل حرف
  - ينتهي بعلامة خاصة "null (\0)" لتحديد نهاية النص

```
string firstName = "Ada";
string lastName = "Lovelace";
string fullName = firstName + " " + lastName;
```

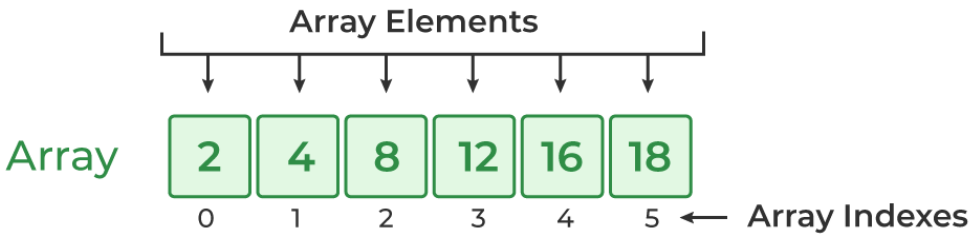
المتغيرات واسناد القيم

- البيانات المنطقية (Boolean) :
  - تستخدم لتخزين القيم المنطقية *True* (1) أو False (0)
  - يتألف غالبا من بت واحد ولكن يخزن عادة في بايت واحد
  - اذا لم يتم تعيين قيمة تعيين f0.0 بشكل افتراضي

```
bool isCodingFun = true;
bool isFishTasty = false;
cout << isCodingFun;  // Outputs 1 (true)
cout << isFishTasty;  // Outputs 0 (false)
```

المتغيرات واسناد القيم

b) الأنواع المركبة (Composite Data Types) :



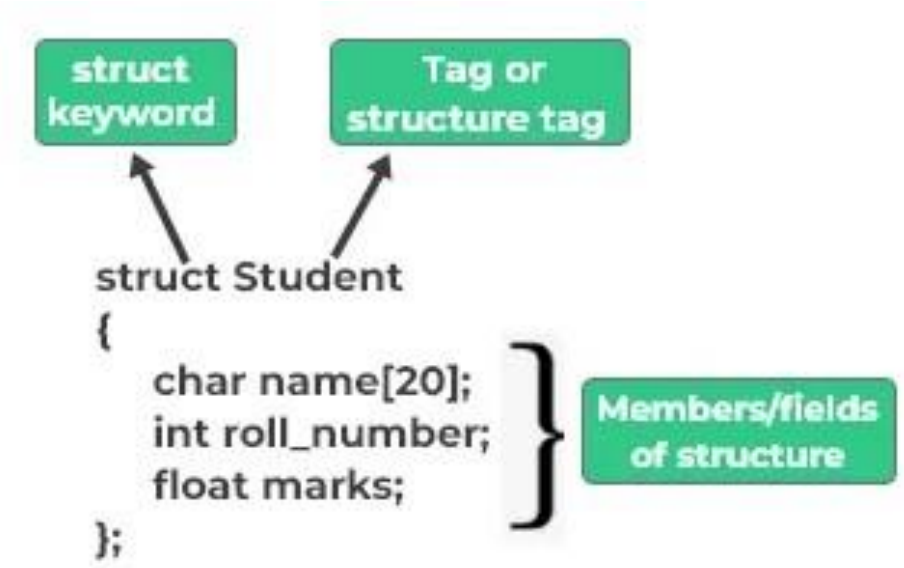
- المصفوفات (Arrays)
  - تستخدم لتخزين مجموعة من القيم المتشابهة في النوع
  - تخزن في مواقع متجاورة في الذاكرة
  - يمكن أن تكون أحادية البعد أو متعددة الأبعاد
  - يمكن الوصول إلى أي عنصر في المصفوفة مباشرة باستخدام الفهرس
  - عند تعريف المصفوفة يتم تحديد حجمها ولا يمكن تغييره أثناء تشغيل البرنامج في معظم اللغات

```
int main()
{
    int array[5] = { 1,7,9,4,5 };
```

```
int main() {
    int arr[3][3] = {{2, 4, 5},{1, 8, 6},{1, 1, 9}};
```

المتغيرات واسناد القيم

- الهياكل (Structures)
  - تستخدم لتجميع بيانات مختلفة الأنواع تحت اسم واحد
  - تستخدم لتوصيف الكائنات المعقدة مثل معلومات الطالب أو معلومات السيارة
  - يمكن الوصول إلى كل عنصر داخل الهيكل باستخدام اسمه
  - يمكن إضافة المزيد من المتغيرات حسب الحاجة



المتغيرات واسناد القيم

- الاتحادات (Unions)

- تستخدم لتوفير نفس مساحة الذاكرة لعدة متغيرات مع القدرة على استخدام واحد منها فقط في وقت معين
- حجم الاتحاد يساوي حجم أكبر عضو داخله
- يتم الوصول إلى الأعضاء باستخدام النقطة (.)

```
3   union university
4   {
5       int s_age;
6       char s_name[30];
7   }
8   main()
9   {
10      union university student;
11      student.s_age=22;
12      printf("Student Age: %d\n", student.s_age);
13      strcpy(student.s_name, "Madhur");
14      printf("Student Name: %s", student.s_name);
```

المتغيرات واسناد القيم

من حيث	الاتحادات	الهياكل
القيمة المخزنة	يمكن تخزين قيمة واحدة فقط في وقت معين	يمكن تخزين قيم لجميع الأعضاء في وقت واحد
الذاكرة	جميع الأعضاء يتشاركون نفس موقع الذاكرة، مما يوفر مساحة	لكل عضو مساحة مستقلة في الذاكرة
المرونة	أقل مرونة بسبب القيود على القيم المخزنة	مرنة لتخزين أنواع بيانات مختلفة
الاستخدام الأساسي	توفير الذاكرة عند الحاجة لتخزين قيمة واحدة فقط	تجميع البيانات المرتبطة ببعضها
الدوال	لا يمكن تضمين الدوال داخل الاتحاد	يمكن تضمين الدوال فقط في C++

المتغيرات واسناد القيم

C) الأنواع المجردة (Abstract Data Types) :

- القوائم (Lists)

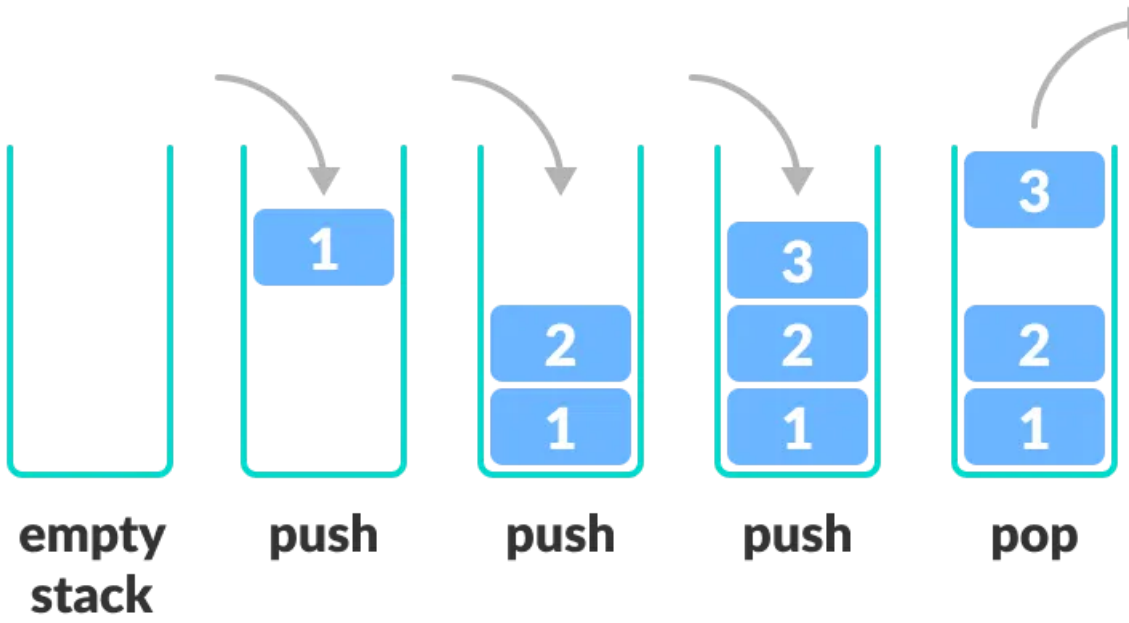
- هي بنية بيانات تستخدم لتخزين مجموعة من العناصر المرتبة
- يتم تخزين العناصر في ترتيب معين ويمكن الوصول إليها بواسطة فهرس
- يمكن أن تحتوي على عناصر من أنواع بيانات مختلفة
- يمكن أن تغير حجمها (إضافة وحذف العناصر)

```
# Define a list of strings
list_1 = ["New York", "Tokyo", "Montreal", "Berlin"]

# List of integers
list_2 = list((1, 5, 8, 9))

# List of booleans
list_3 = [True, False, False, True]

# Mixed list
list_4 = list((True, 25, False, "hello"))
```

المتغيرات واسناد القيم

- المكدسات (Stacks)

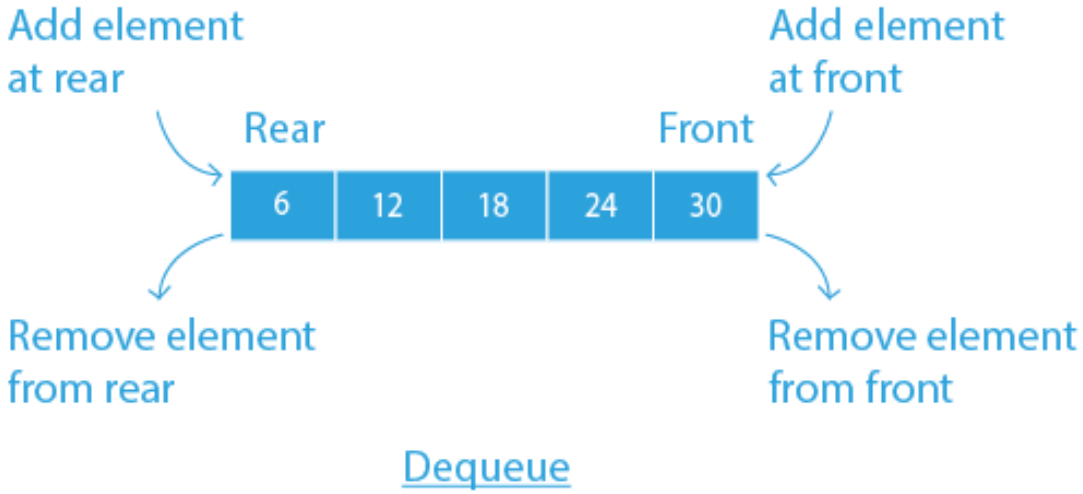
- هو نوع من بنى البيانات يعمل وفق مبدأ Last In, First Out (LIFO)، أي أن العنصر الأخير الذي يدخل هو أول عنصر يخرج
- لا يمكن الوصول إلى العناصر إلا من الأعلى



المتغيرات واسناد القيم

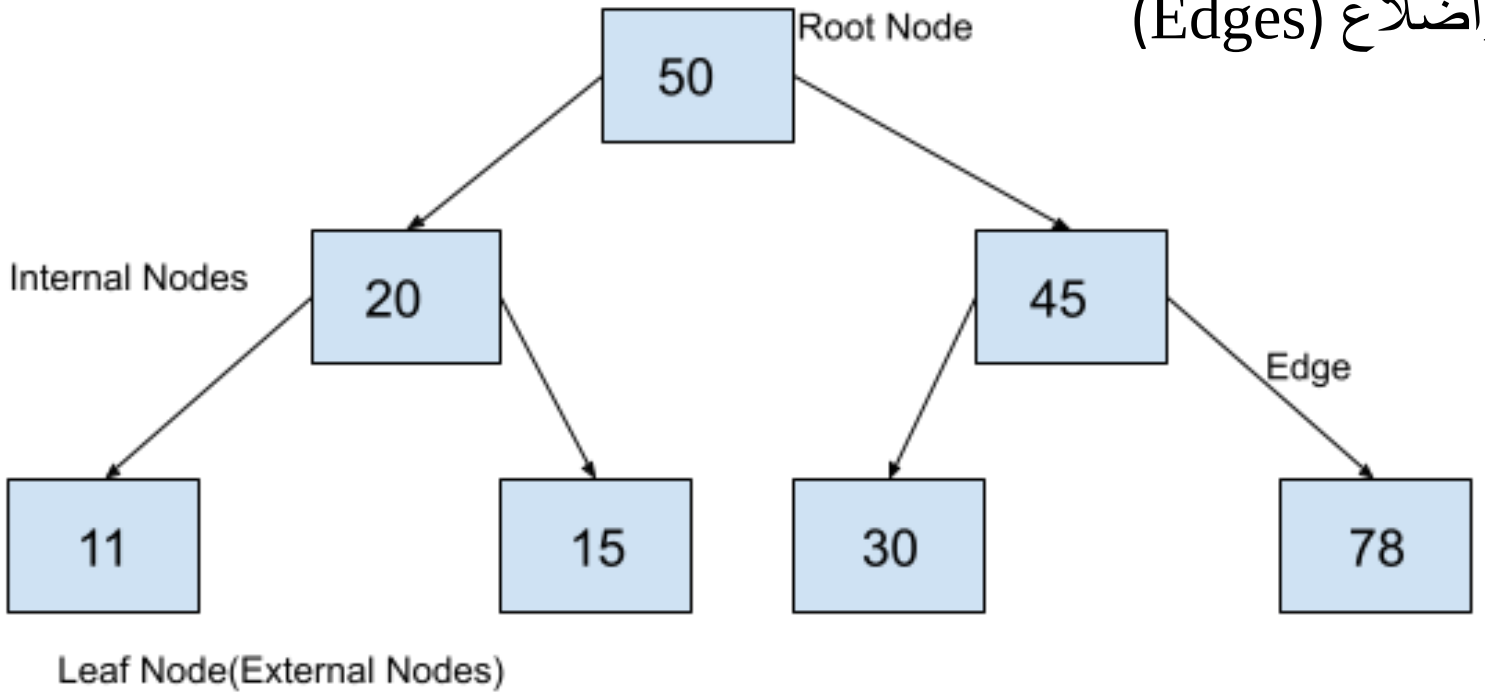
- الطوابير (Queues)

- هو نوع من بنى البيانات يعمل وفق مبدأ First In, First Out (FIFO)، أي أن العنصر الذي يدخل أولا هو أول عنصر يخرج
- يمكن الإضافة فقط من النهاية والحذف من البداية



المتغيرات واسناد القيم

- الأشجار (Trees)
  - هي بنية بيانات هرمية تستخدم لتمثيل البيانات بطريقة شبيهة بفروع الشجرة الطبيعية
  - لشجرة تتكون من عقد (Nodes) وأضلاع (Edges)



تخزين البيانات

1 - أنواع وسائط التخزين (Types of storage) :

a) الذاكرة الأساسية (Primary Memory) **-** RAM **:** تستخدم لتخزين البيانات أثناء تنفيذ البرامج و توفر سرعة وصول عالية جدا مقارنة بوسائط التخزين الثانوية لكنها تفقد البيانات عند انقطاع الطاقة

b) التخزين الثانوي (Secondary Storage):

- يستخدم لتخزين البيانات بكميات كبيرة وبشكل دائم مثل :

- الأشرطة المغناطيسية (Magnetic Tapes)
- الأقراص المغناطيسية (Magnetic Disks)
- الأقراص الضوئية (Optical Disks)
- الذاكرة الصلبة (Solid-State Memory)

تخزين البيانات

من حيث	الأشرطة المغناطيسية	الأقراص المغناطيسية	الأقراص الضوئية	الذاكرة الصلبة
السرعة	بطيئة جدا تعتمد على الوصول التسلسلي	سرعة عالية تعتمد على الوصول العشوائي	سرعة متوسطة تعتمد على الليزر للقراءة	عالية جدا تعتمد على الدوائر الإلكترونية
السعة التخزينية	كبيرة جدا	كبيرة جدا	متوسطة	متوسطة إلى كبيرة
الاستخدام	النسخ الاحتياطي طويل الأمد	تخزين البيانات اليومية في الحواسيب المكتبية والخوادم	تخزين الوسائط المتعددة مثل الموسيقى والأفلام	التخزين السريع في الأجهزة الحديثة كالحواسيب المحمولة
الأداء	أداء منخفض بسبب سرعة الوصول البطيئة	أداء عال وسرعة استجابة جيدة	أداء معتدل مع استخدام القراءة المتتابعة	أداء فائق السرعة مع نقل البيانات بسرعة عالية

تخزين البيانات

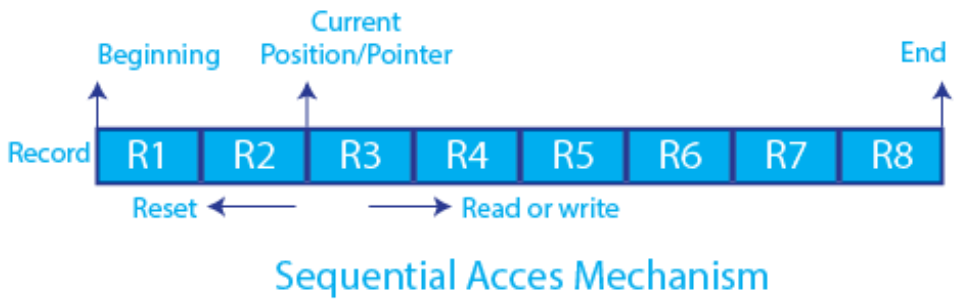
2 - السجلات (Records) : السجل هو مجموعة مرتبطة من العناصر البيانية (Data Items) التي تصف كيانا معينا بشكل شامل و يتكون من عدة حقول (Fields) مثل سجل موظف يحتوي على [الراتب,الاسم, الرقم التعريفي]

- التنظيم : يقوم بتسهيل تخزين البيانات بشكل مرتب ومنظم
- الوصول السريع : يمكن البحث عن سجل معين بسرعة
- التكرار : يمكن تكرار الهيكل نفسه لجميع السجلات بسهولة
- التكامل : الحفاظ على البيانات المرتبطة معا

تخزين البيانات

3 - تنظيم الملفات (File Organizing) : يوجد العديد من الانواع لتنظيم الملفات :

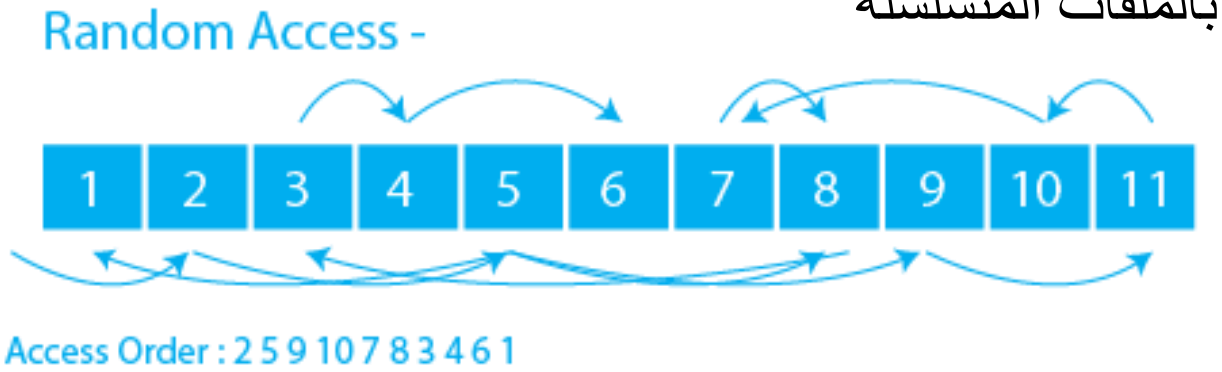
- الملفات المتسلسلة (Sequential Access Files) :
  - يتم الوصول إلى البيانات بترتيب تسلسلي
  - لتحديد سجل معين يجب قراءة جميع السجلات التي تسبقه
  - مناسب للبيانات التي يتم معالجتها بشكل متسلسل مثل ملفات النسخ الاحتياطي
  - بسيط وسهل الاستخدام
  - بطيء في الوصول إلى السجلات المحددة



Sequential Acces Mechanism

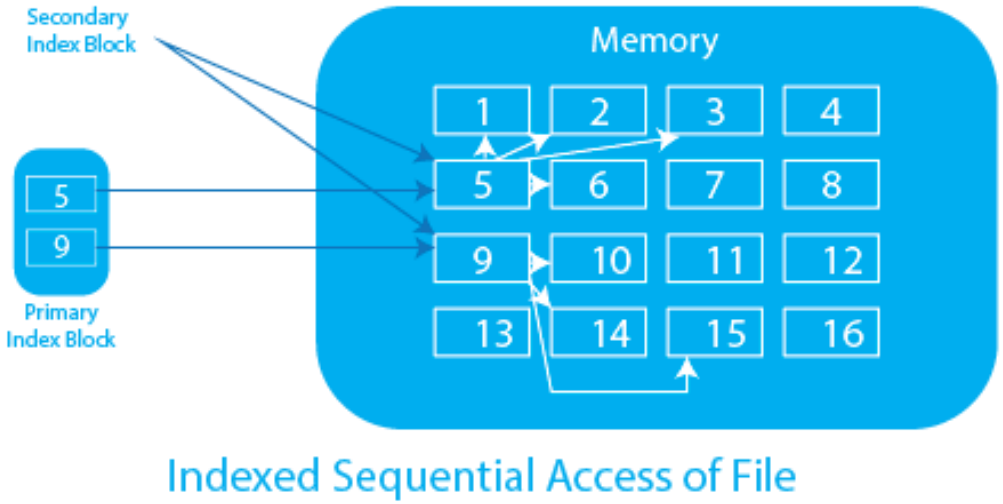
تخزين البيانات

- الملفات العشوائية (Random Access Files) :
  - يمكن الوصول إلى أي سجل مباشرة دون الحاجة لقراءة السجلات السابقة
  - يتم تحديد موقع السجل باستخدام عنوان أو مؤشر
  - مناسب للأنظمة التي تتطلب وصولا سريعا مثل أنظمة الحجز
  - سرعة كبيرة في الوصول إلى السجلات
  - أكثر تعقيدا في التنظيم مقارنة بالملفات المتسلسلة



تخزين البيانات

- الملفات المتسلسلة المفهرسة (Indexed Sequential Access Files) :
  - يتم استخدام فهرس لتحديد موقع السجل بسرعة مع الاحتفاظ بترتيب تسلسلي للسجلات
  - شائع في التطبيقات التي تحتاج إلى معالجة تسلسلية وعشوائية مثل الأنظمة المصرفية
  - سرعة في الوصول إلى البيانات
  - الاحتفاظ بترتيب منطقي للبيانات
  - يحتاج إلى وقت وجهد أكبر لإدارة الفهرس



Indexed Sequential Access of File

تخزين البيانات

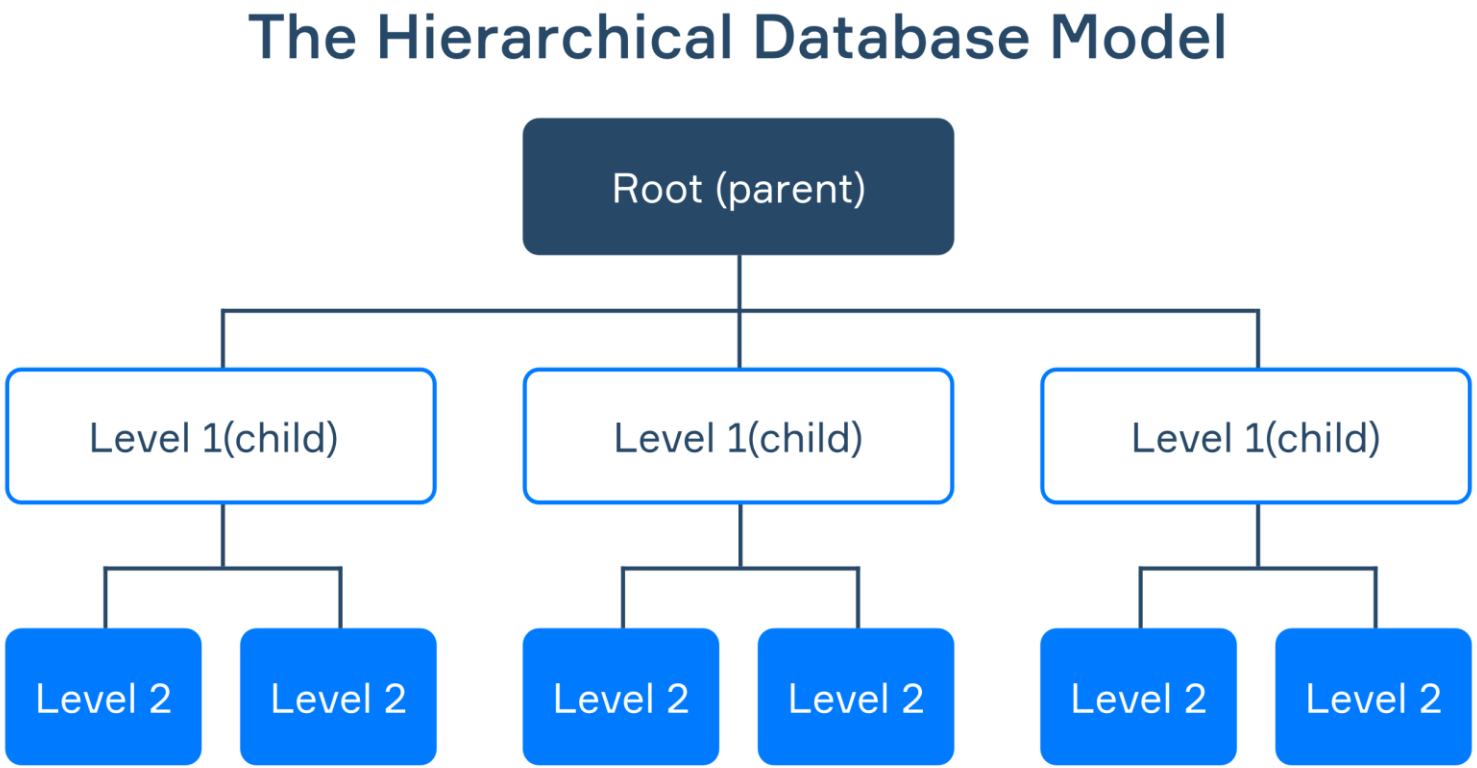
4 - نظم إدارة قواعد البيانات (DBMS)

الفرق بين الملفات البسيطة وقواعد البيانات

من حيث	تنظيم الملفات	قواعد البيانات
المرونة	تغيير البنية يتطلب تعديل جميع البرامج المرتبطة	يمكن تعديل الجداول دون الحاجة لتعديل البرامج
إدارة التكرار	لا توجد إدارة للتكرار ما يؤدي إلى ازدواجية البيانات	التكرار مدار من خلال العلاقات بين الجداول
الأداء	بطيء عند البحث عن البيانات بسبب غياب الفهارس	أسرع بسبب دعم الفهارس والتخزين الهيكلي
الأمان	بيانات أقل أمانا وصعوبة في إدارة الوصول	أدوات أمان متقدمة مثل إدارة المستخدمين والصلاحيات

تخزين البيانات

- قواعد البيانات الهرمية (Hierarchical DB) : تنظم البيانات في شكل شجرة حيث لكل سجل رئيسي عدة سجلات فرعية



تخزين البيانات

- قواعد البيانات الشبكية (Network DB) : تنظم البيانات في شكل شجرة حيث لكل سجل رئيسي عدة سجلات فرعية

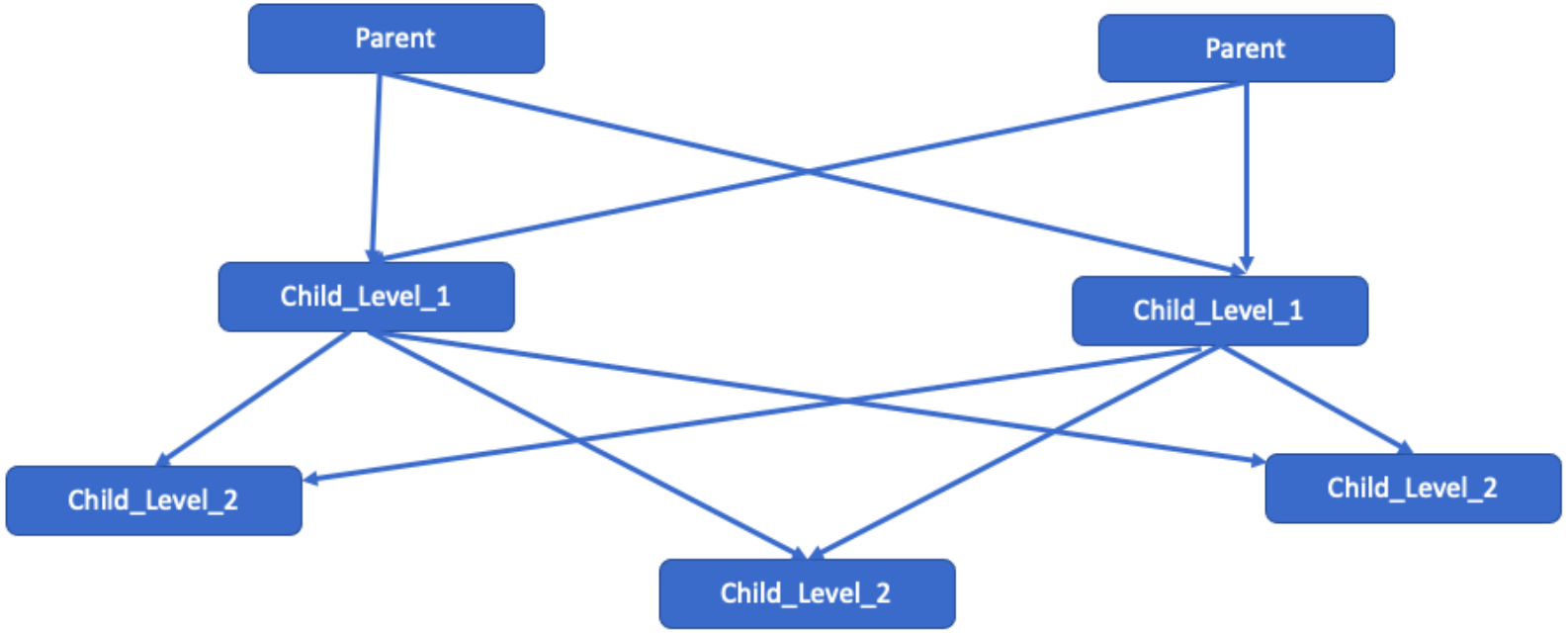


Fig: Network Database Model

تخزين البيانات

- قواعد البيانات العلائقية (Relational DB) : تعتمد على الجداول والعلاقات البسيطة بينها مما يجعلها الأكثر شيوعا وسهولة في الاستخدام

id	name	type	size
1	Rex	dog	big
2	Cleo	dog	small
3	Leroy	dog	medium
4	Sunny	cat	big
5	Snow Ball	cat	medium

id_pet	weight	age
1	32,3	7
2	4,35	13,6
3	24,9	4
5	8,75	5

name	food	danger_level
Cleo	diet	5
Sunny	wet	1
Rex	dry	4
Leroy	wet	3

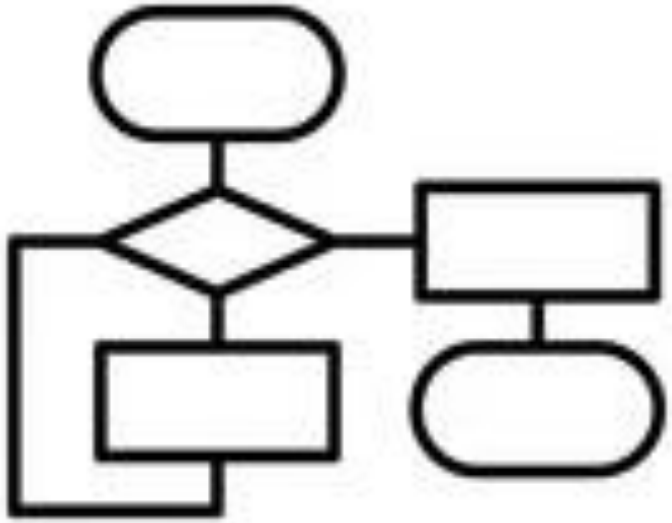
الخوارزميات والمخططات الانسيابية

1 - الخوارزميات (Algorithms) : الخوارزمية هي مجموعة مرتبة من التعليمات أو الخطوات المنطقية التي تُنفذ بشكل متسلسل لحل مشكلة معينة أو لتحقيق هدف محدد و توفر هيكلا منظما يضمن تنفيذ العمليات بشكل صحيح وفعال

- الوضوح : تسهل فهم المشكلة وخطوات الحل
- التكرار : يمكن تنفيذ الخوارزمية نفسها لحل المشكلات المتشابهة
- الكفاءة : تتيح تحسين الأداء وتقليل الزمن اللازم للحل

الخوارزميات والمخططات الانسيابية

2 - المخططات الانسيابية (Flowcharts) : المخطط الانسيابي هو تمثيل بصري للخوارزمية باستخدام رموز معيارية متصلة بأسهم توفر رؤية شاملة ومنظمة للخطوات و تساعد في شرح الخطوات للمبرمجين أو الأشخاص غير التقنيين ولها عدة قواعد و رموز :



- قواعد تصميم المخططات الانسيابية :
  - البداية والنهاية : تكون بدائرة بيضاوية (End, Start)
  - الاتجاه : تدفق العملية يكون عادة من الأعلى إلى الأسفل، ومن اليسار إلى اليمين
  - التقسيم : يمكن تقسيم المخطط إلى صفحات إذا كان معقدا باستخدام رموز الموصلات

الخوارزميات والمخططات الانسيابية

- رموز المخططات الانسيابية الأساسية :

النماذج او التقارير

Document / Report

نماذج المدخلات والطلبات او

نماذج وتقارير المخرجات

مثال لمدخل (نموذج طلب اجازة)

مثال لمخرج (نموذج شهادة تعريف)

وتقارير اجازات الموظفين

اتخاذ القرارات

decision

وعادة مايتم الكتابة بداخله على شكل

سؤال والاجابة تكون بمسارين

نعم او لا

كتابة

المدخلات / المخرجات

DATA

Input/output

وتعرف بأنها مدخلات للعمليات

ومخرجات من العمليات

للبدء في سلسة العمليات او انهائها

START/END

استدعاء اجراء آخر

Call-Process

كل اجراء عبارة عن عمليات

واجراءات اخرى لدى جهات مختلفة

يتم استخدام هذا الرمز عند

استدعاء اجراء لدى جهة اخرى منفصلة

عن الجهة الحالية

التوصيل بين الصفحات

Off-Page

بعض الاجراءات ويلة وتحتاج

الى عدة صفحات

وبالتالي فان هذا الرمز يسهل

عملية التنقل

والربط بين صفحات الاجراء نفسه

التوصيل بين العمليات البعيدة

Connector

وهذا الرمز يستخدم لتسهيل عمليات الربط

في الاجراؤات الطويلة والمعقدة

وذلك لايصال الاجراء للقارئ بصورة

اسهل

كتابة العملية

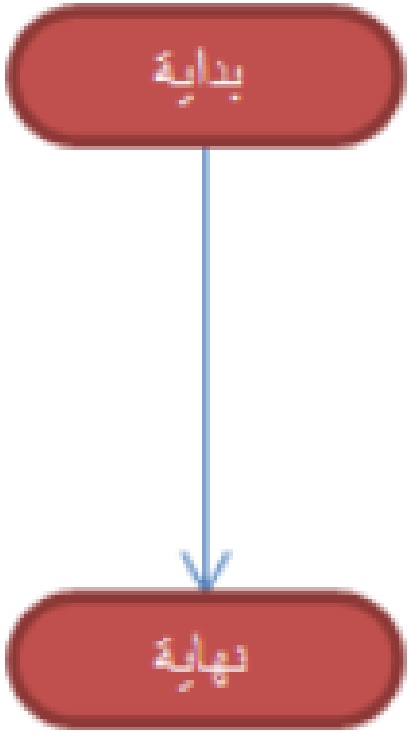
Process

وتعرف بالافعال والانشطة التي تؤدي

الى وجود مخرجات

الخوارزميات والمخططات الانسيابية

❖ امثلة عن رموز المخططات الانسيابية الأساسية :



للبدء في سلسة العمليات او انهائها

START/END

الخوارزميات والمخططات الانسيابية

كتابة العملية

Process

وتعرف بالافعال والانشطة التي تؤدي

الى وجود مخرجات

الخوارزميات والمخططات الانسيابية

كتابة

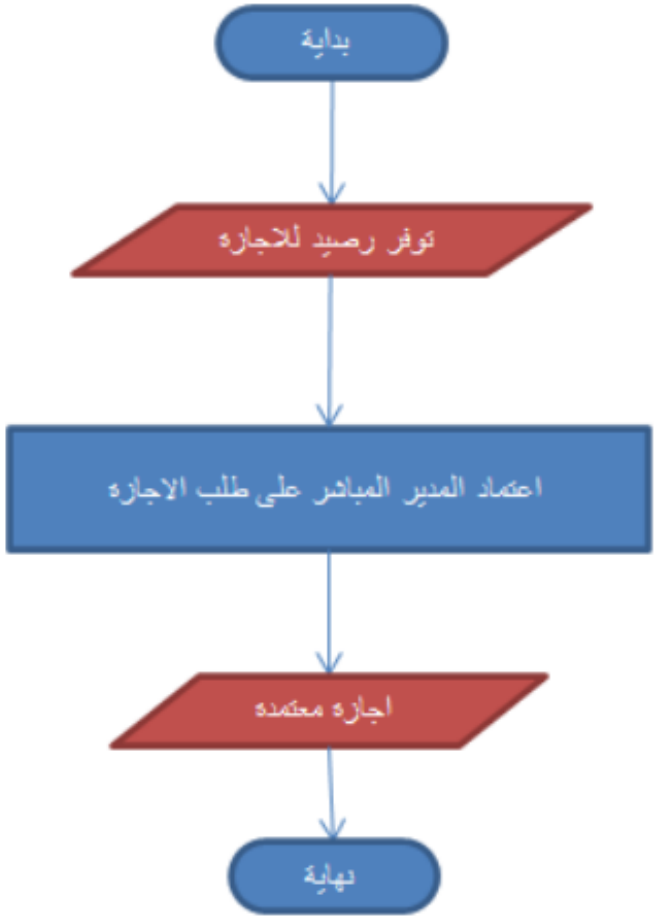
المدخلات / المخرجات

DATA

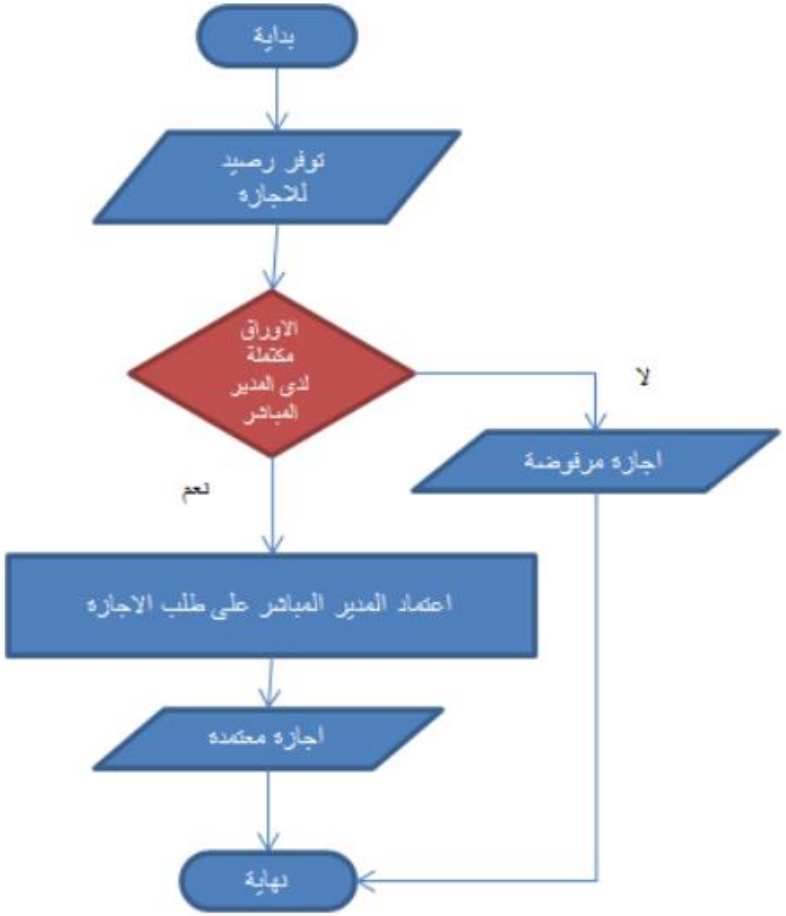
Input/output

وتعرف بأنها مدخلات للعمليات

ومخرجات من العمليات



الخوارزميات والمخططات الانسيابية



اتخاذ القرارات

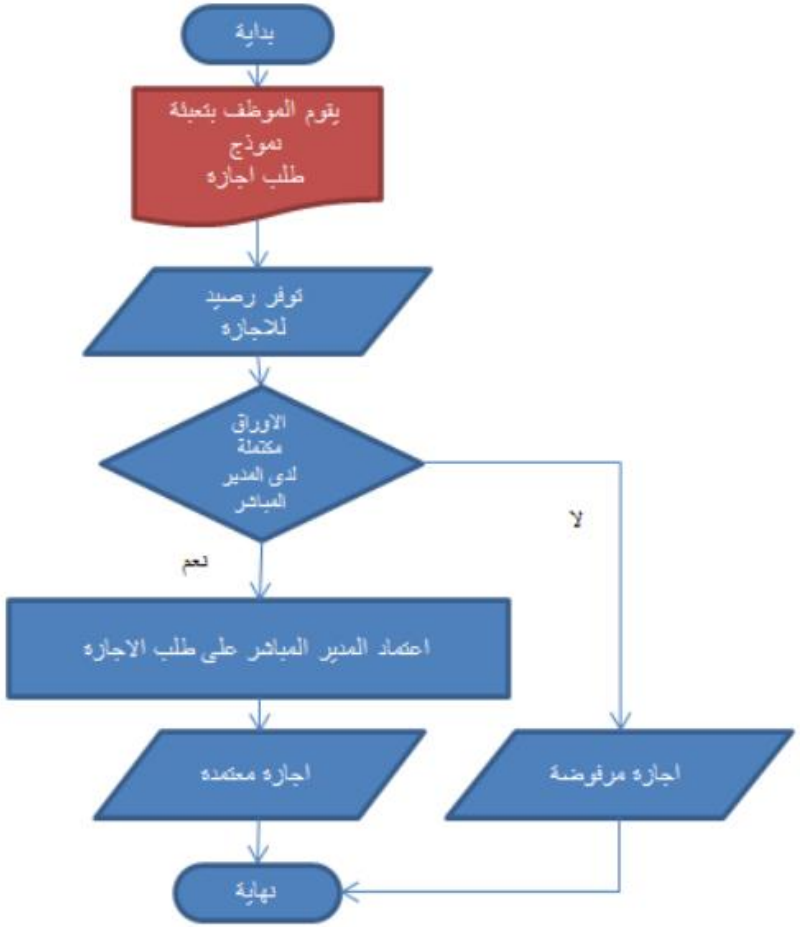
decision

وعادة مايتم الكتابة بداخله على شكل

سؤال والاجابة تكون بمسارين

نعم او لا

الخوارزميات والمخططات الانسيابية



النماذج او التقارير

Document / Report

نماذج المدخلات والطلبات او

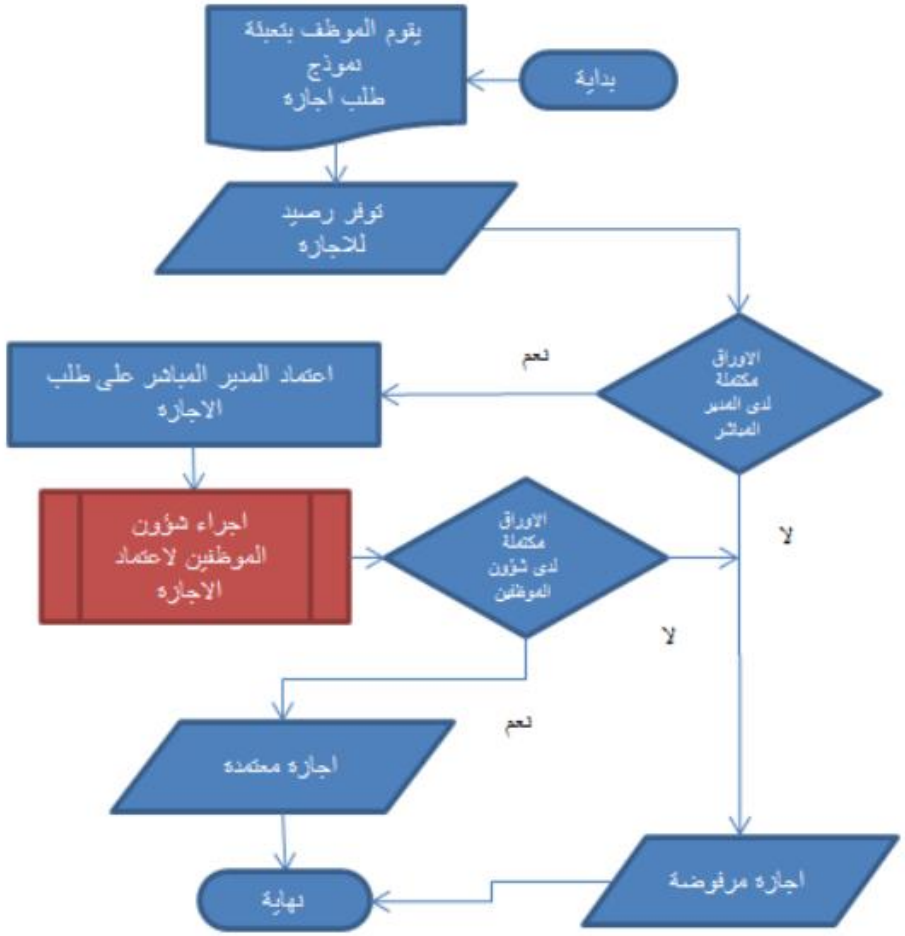
نماذج وتقارير المخرجات

مثال لمدخل (نموذج طلب اجازة)

مثال لمخرج (نموذج شهادة تعريف)

وتقارير اجازات الموظفين

الخوارزميات والمخططات الانسيابية



استدعاء اجراء آخر

Call-Process

كل اجراء عبارة عن عمليات

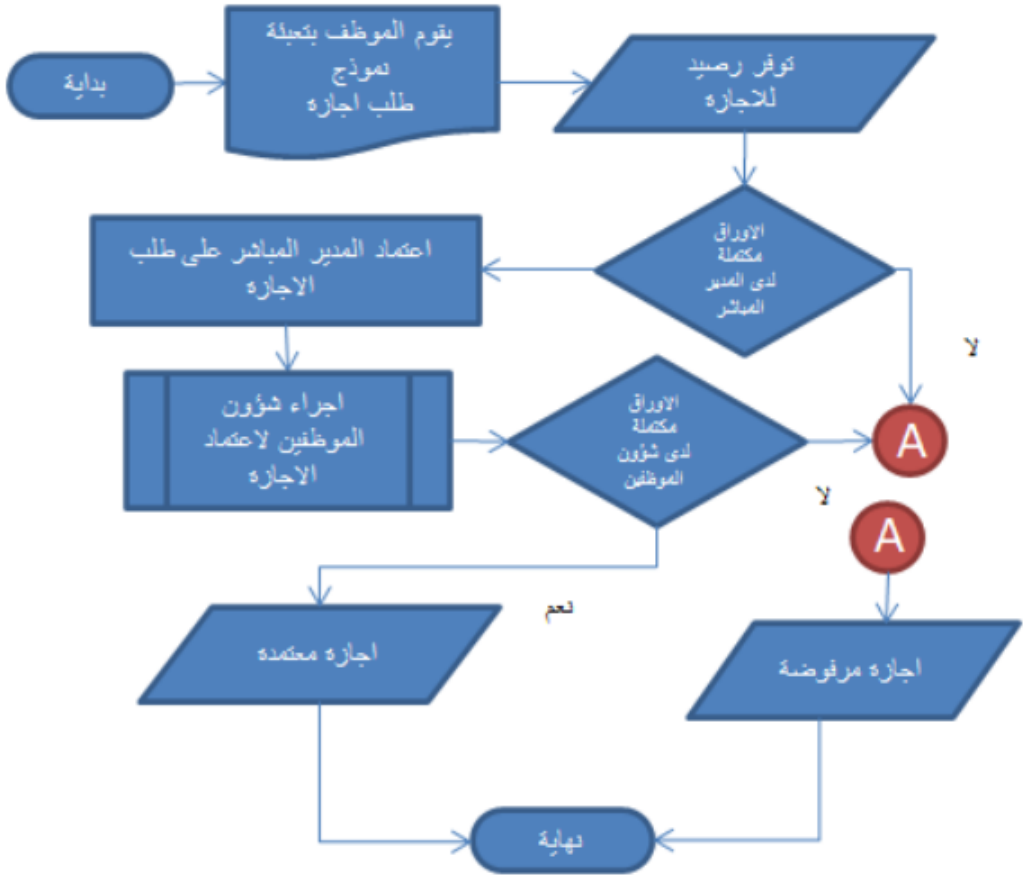
واجراءات اخرى لدى جهات مختلفة

يتم استخدام هذا الرمز عند

استدعاء اجراء لدى جهة اخرى منفصلة

عن الجهة الحالية

الخوارزميات والمخططات الانسيابية

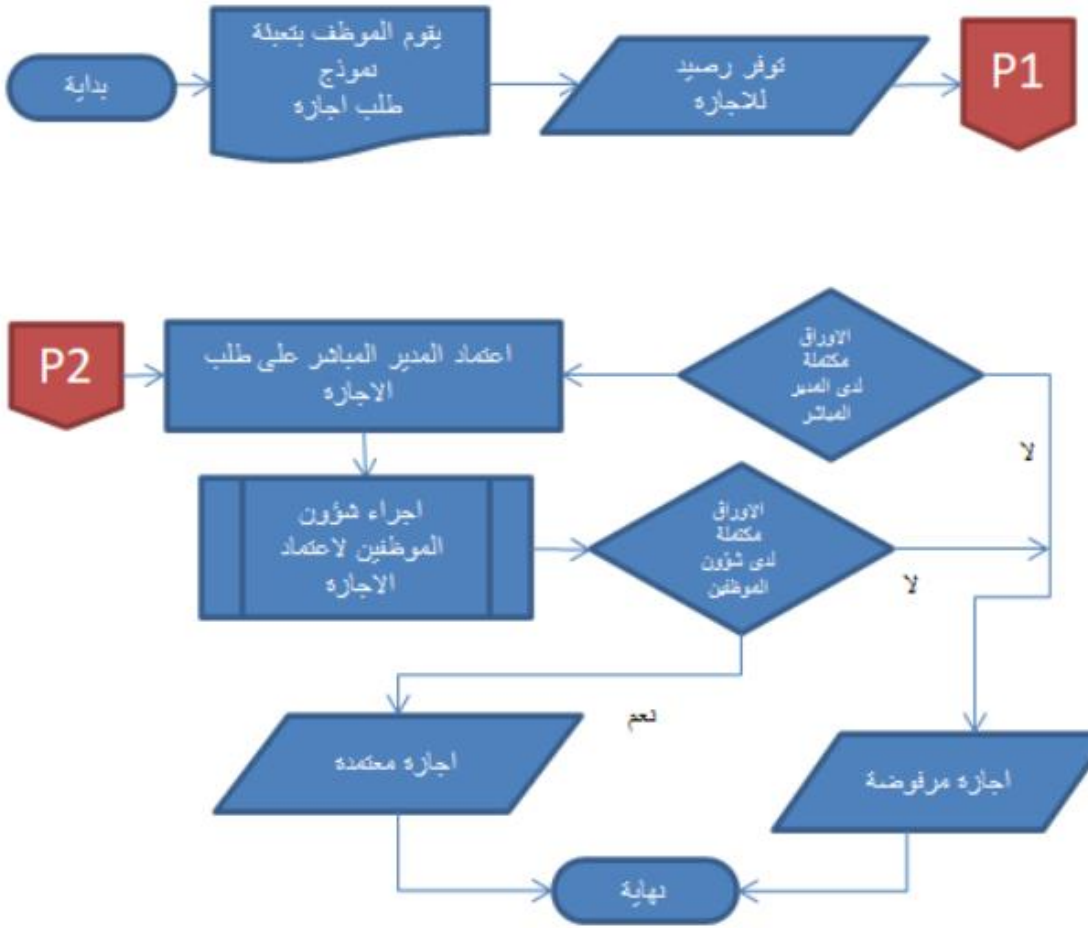


التوصيل بين العمليات البعيدة

Connector

وهذا الرمز يستخدم لتسهيل عمليات الربط في الاجراؤات الطويلة والمعقدة وذلك لايصال الاجراء للقارئ بصورة اسهل

الخوارزميات والمخططات الانسيابية



التوصيل بين الصفحات

Off-Page

بعض الاجراءات وبلة وتحتاج

الى عدة صفحات

وبالتالي فان هذا الرمز يسهل

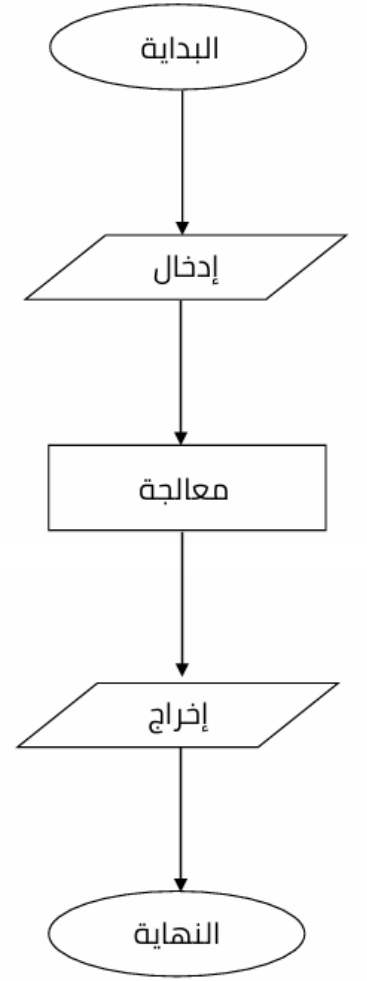
عملية التنقل

والربط بين صفحات الاجراء نفسه

الخوارزميات والمخططات الانسيابية

- انواع المخططات الانسيابية :
- **خرائط التتابع البسيطة (Simple Sequential Flowcharts) :**

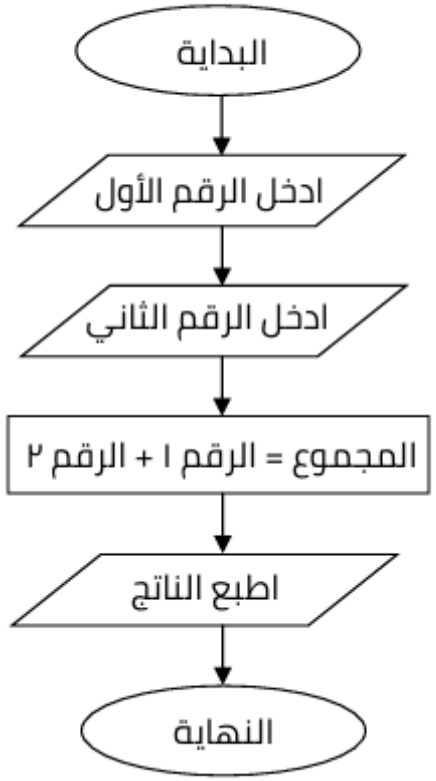
يتم ترتيب الخطوات لهذا النوع على شكل سلسلة مستقيمة من البداية للنهاية بدون تفرعات او دوران



الخوارزميات والمخططات الانسيابية

❖ مثال :

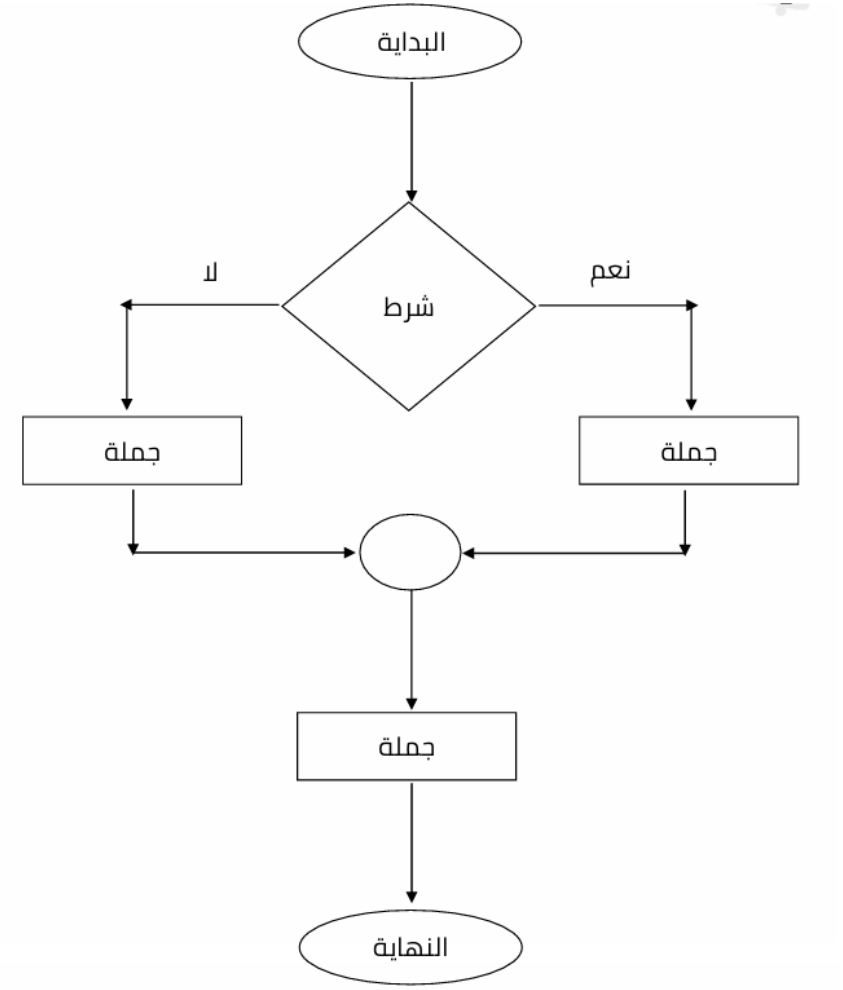
كتابة الخوارزمية والمخطط الانسيابي لبرنامج يقوم بجمع عددين ثم يقوم بطابعة الناتج

طريقة الخوارزميات	طريقة خرائط التدفق
• البداية . • ادخل الرقم الأول Num1 • ادخل الرقم الثاني Num2 • المجموع = الرقم الأول + الرقم الثاني Sum = Num1 + Num2 • اطبع الناتج Print Sum • النهاية	

الخوارزميات والمخططات الانسيابية

- **خرائط ذات الفروع (Branched Flowcharts) :**

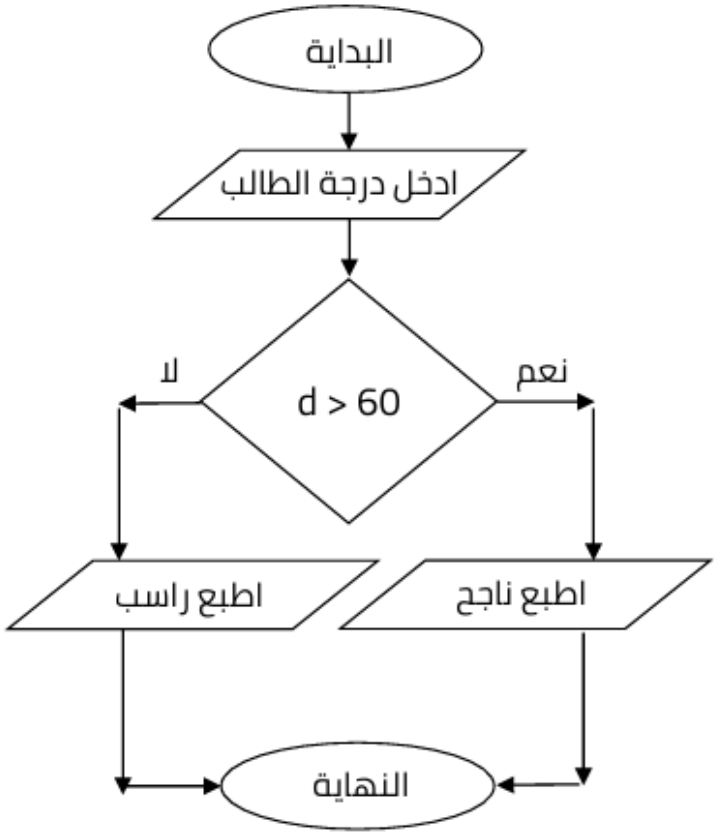
تعبر عن الخرائط ذات التفرعات بسبب الحاجة لأتخاذ قرار او مفاضلة بين اختياريين او اكثر فيسير كل اختيار في تفرع مستقل



الخوارزميات والمخططات الانسيابية

❖ مثال :

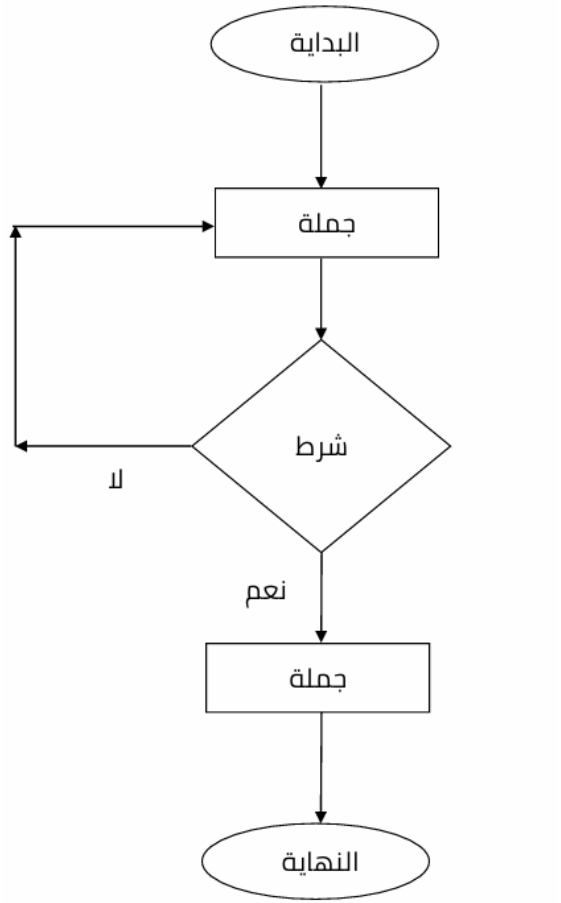
كتابة الخوارزمية والمخطط الانسيابي لبرنامج يقوم بقراءة درجة الطالب في الاختبار في حال كانت الدرجة اكبر او تساوي ال 60 اكتب ناجح اما في حال كانت اقل اكتب راسب

طريقة الخوارزميات	طريقة خرائط التدفق
• البداية • ادخل درجة الطالب input d • قم بمقارنة الدرجة مع 60 • إذا كانت الدرجة أكبر من أو يساوي 60 اطبع كلمة ناجح • إذا كانت الدرجة أصغر من 60 اطبع كلمة راسب • النهاية	

الخوارزميات والمخططات الانسيابية

- **خرائط الدوران البسيط (Simple - loop Flowcharts) :**

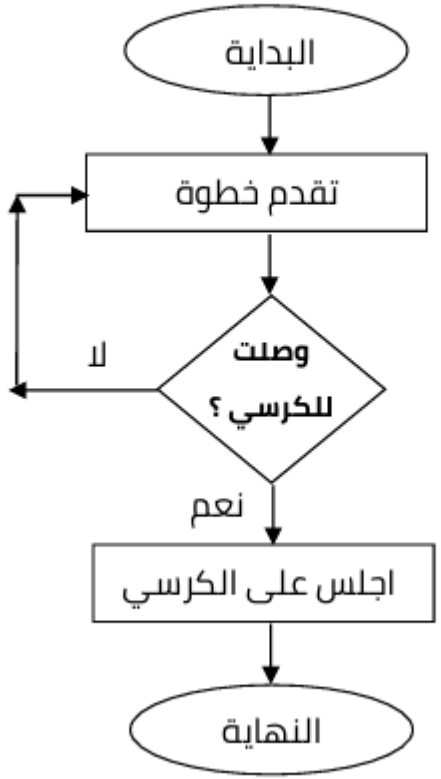
نستخدم هذه الخرائط للقيام بتكرار عملية او مجموعة عمليات في البرنامج مرات عديدة



الخوارزميات والمخططات الانسيابية

❖ مثال :

كتابة الخوارزمية والمخطط الانسيابي لشخص يريد الجلوس على كرسي يبعد عنه عدة امتار

طريقة الخوارزميات	طريقة خرائط التدفق
• البداية • تقدم خطوة • اذا لم تصل الى الكرسي انتقل الى الخطوة رقم 2 • اذا وصلت الى الكرسي انتقل الى الخطوة رقم 4 • اجلس على الكرسي • النهاية	

الخوارزميات والمخططات الانسيابية

امثلة عن استخدام المخططات الانسيابية في البرمجة :

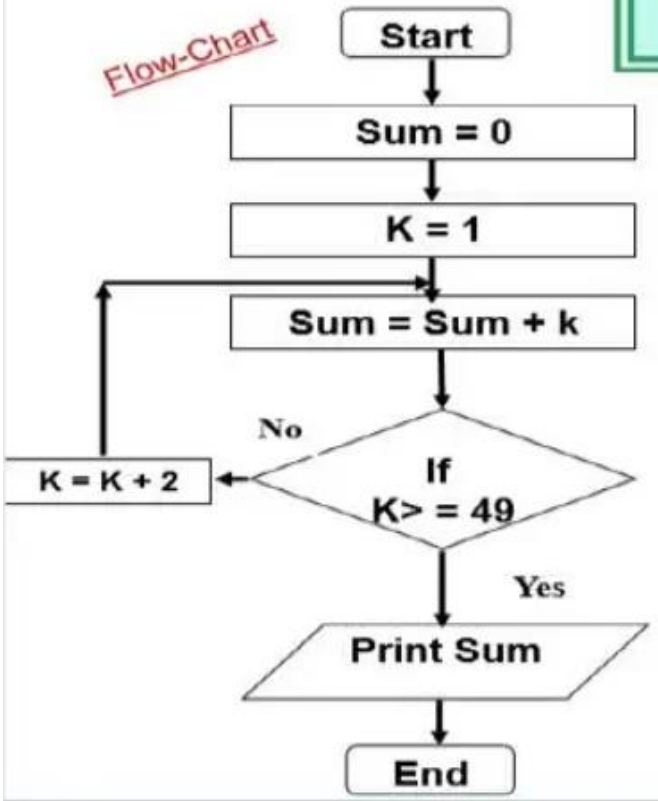
مثال

مثال: حساب مجموع بعض الأعداد:

ارسم خريطة التدفق لحساب حاصل جمع أول 25 عدد فردي من الأعداد الصحيحة كالاتي.

Sum = 1 + 3 + 5 + 7 +.....49

Flow-Chart



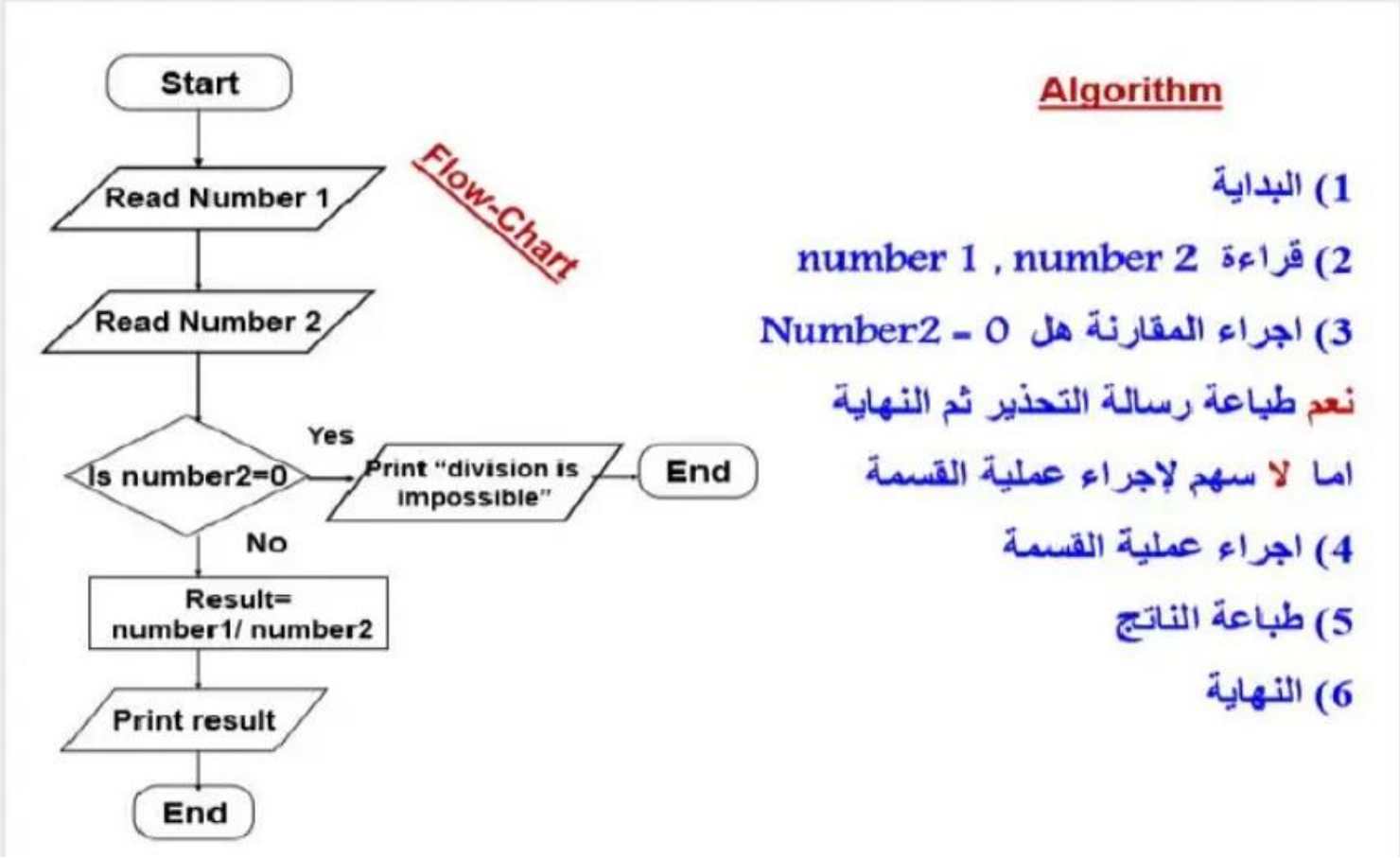
Algorithm

1. Start
2. Let sum=0 and k=1
3. Sum=sum+k;
4. If (k>=49) Then go to (7)
5. k=k+2;
6. go to (3)
7. Print sum
8. End

الخوارزميات والمخططات الانسيابية

❖ مثال :

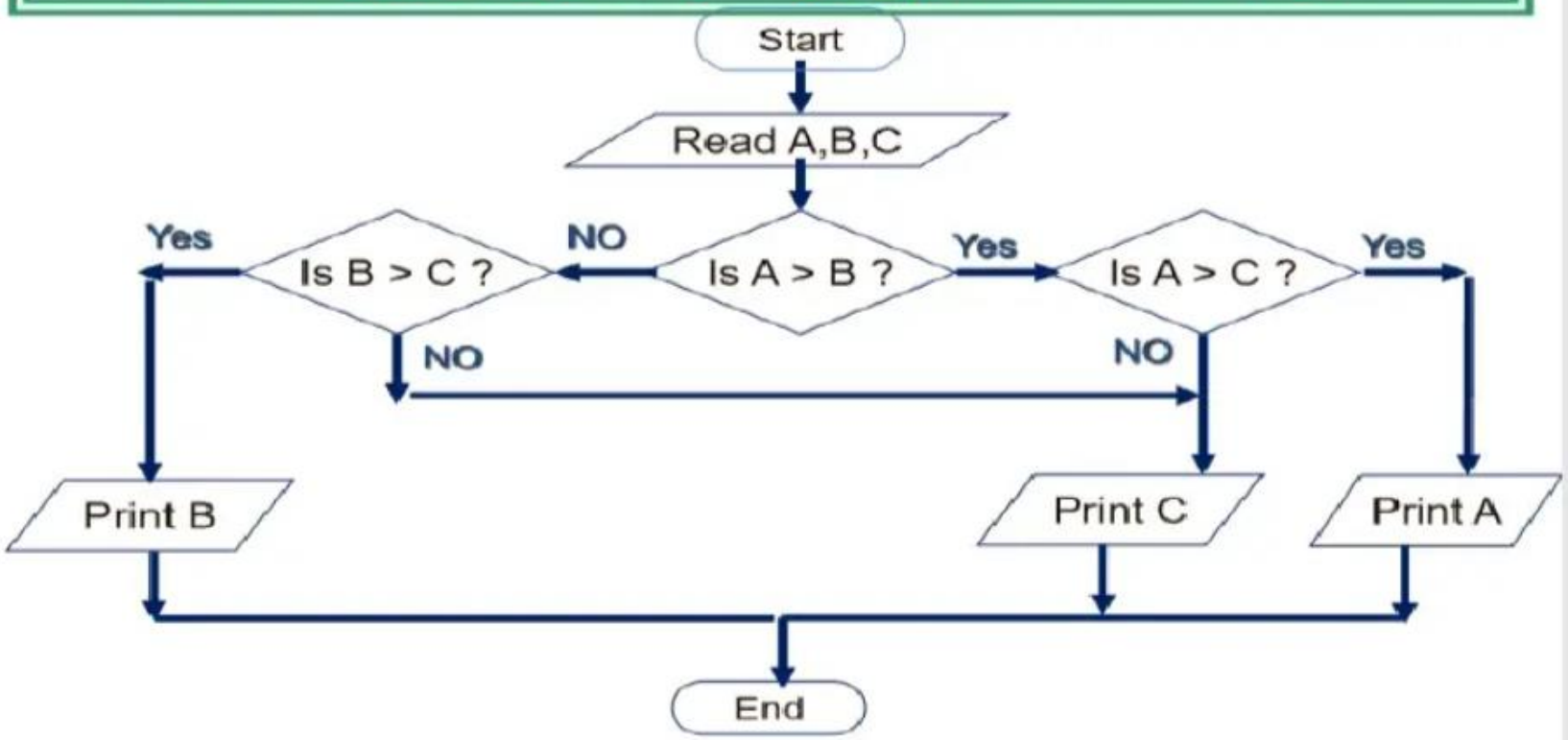
كتابة الخوارزمية والمخطط الانسيابي لبرنامج يقوم بقراءة رقميين ويقوم بعملية قسمة الرقم الاول على الثاني وارسال رسالة في حال كان الرقم الثاني صفر



الخوارزميات والمخططات الانسيابية

مثال

ارسم المخطط الانسيابي لإيجاد الرقم الأكبر من بين ثلاثة أرقام



الخوارزميات والمخططات الانسيابية

مثال

حساب الاجور المستحقة للكهرباء

ارسم خارطة التدفق تمثل حساب الاجور المستحقة للكهرباء علما بأن :

القراءة الحالية (A)

القراءة السابقة (B)

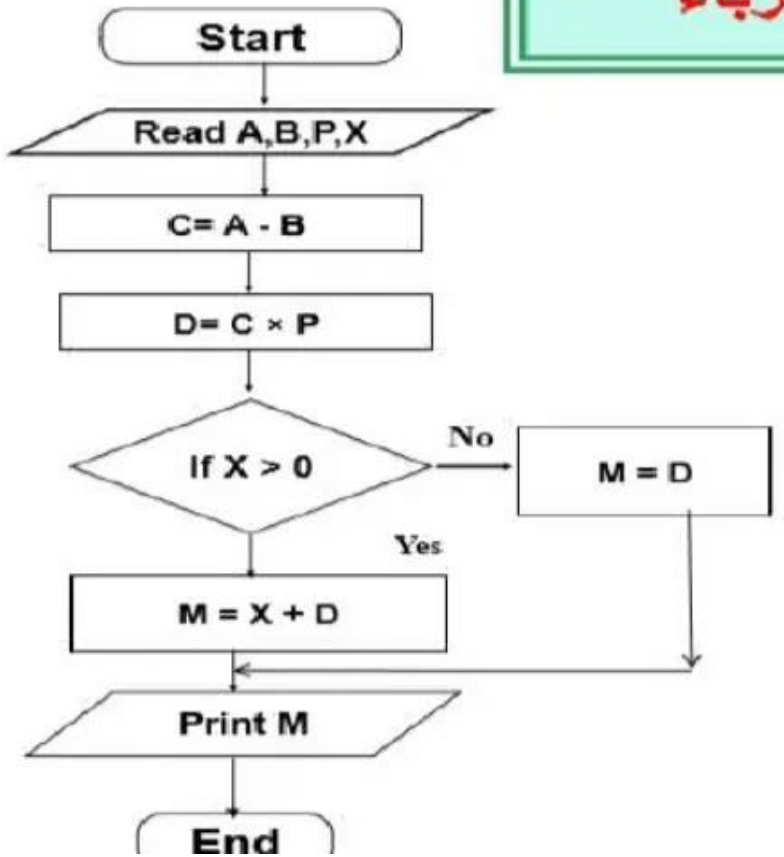
سعر الوحدة (P)

الديون السابقة (X)

الاستهلاك (C)

الأجور (D)

الاجور الكلية الواجب دفعها (M).



الخوارزميات والمخططات الانسيابية

3 - التعامل مع البيانات في الخوارزميات (Data handling in algorithms) :

- القراءة من الملفات : يتم إدخال البيانات يدويا من خلال لوحة المفاتيح او يتم استخدام أوامر برمجية لفتح الملفات وقراءة البيانات منها

```
FILE *file = fopen("data.txt", "r");
fscanf(file, "%d", &value);
fclose(file);
```

- قراءة البيانات من قواعد البيانات : يتم استخدام لغات استعلام مثل SQL للوصول إلى البيانات

```
SELECT * FROM employees WHERE age > 30;
```

- كتابة البيانات : يتم استخدام أوامر برمجية لكتابة البيانات إلى الملفات او الى قواعد البيانات

```
FILE *file = fopen("output.txt", "w");
fprintf(file, "Result: %d", result);
fclose(file);
```

```
INSERT INTO employees (name, age, salary) VALUES ('John Doe', 35, 50000);
```

التعبيرات الحسابية والعلائقية والمنطقية

التعبيرات (Expressions) : التعبيرات في البرمجة تمثل أساسا للتعامل مع البيانات وحل المشكلات يتم استخدامها في معظم العبارات البرمجية وتعتبر جزءا لا يتجزأ من كتابة التعليمات البرمجية و كل تعبير يمثل مجموعة من القيم، المتغيرات، الثوابت، والعوامل المرتبطة باستخدام العوامل الحسابية، المنطقية، أو المقارنة.

- مكونات التعبير : قد يتكون التعبير من متغيرين أو أكثر أو ثابت أو مزيج بينهما ويتم الجمع بين المكونات باستخدام العوامل (Operators)
- التقييم : يتم تقييم التعبير وفقا لقواعد العمليات الحسابية، المقارنة، أو المنطقية المعتمدة في اللغة البرمجية
- التعبيرات تستخدم دائما في الجانب الأيمن من رمز الإسناد (=)

التعبيرات الحسابية والعلائقية والمنطقية

1 - التعبيرات الحسابية (Arithmetic Expressions) : تستخدم لإجراء العمليات الرياضية منها :

- المتغيرات والثوابت الرقمية
- العوامل الحسابية (+,-,*,/,^)
- الأقواس لتنظيم الأولويات
- يمكن استخدام مكتبات رياضية للعمليات المعقدة مثل الجذر التربيعي

a + b

a * (b + c)

(x + y) ^ 2 – z

sqrt(x) + y / z

التعبيرات الحسابية والعلائقية والمنطقية

2 - التعبيرات العلائقية (Relational Expressions) : تستخدم لمقارنة القيم ترجع نتيجة منطقية (Boolean) :

- تتضمن العوامل العلائقية (<,>,<=,>=,==,=!)

a > b

3 - التعبيرات المنطقية (Logical Expressions) : تستخدم لاتخاذ قرارات بناء على عدة شروط :

- تتضمن العوامل المنطقية

x == y

- && : (AND)تتحقق فقط إذا كانت جميع الشروط صحيحة

score >= 50

- || : (OR)تتحقق إذا كان أي شرط صحيح

(a + b) < c

- ! : (NOT)عكس القيمة

(x > y) && (z < 10)

!(a == b)

(score >= 50) || (attendance >= 75)

التعبيرات الحسابية والعلائقية والمنطقية

الرمز	الوظيفة	الأسبقية
!	النفي	1
&&	عملية منطقية (and)	2
||	عملية منطقية (or)	3

الرمز	الوظيفة	الأسبقية
++ أو--	الزيادة والنقصان	1
-	الإشارة السالبة	2
*أو / أو %	الضرب والقسمة وباقيها	3
+ أو -	الجمع أو الطرح	4
=	التساوي	5
++ أو – المتأخرة بعد الرمز	زيادة أو نقصان للعدد	6

الرمز	الوظيفة	الأسبقية
~	إشارة النفي	1
>> أو <<	إزاحة للعدد يمين أو يسار	2
&	عملية (and)	3
^	الرفع لقوى	4
|	عملية (or)	5

التعبيرات الحسابية والعلائقية والمنطقية

العملية	لفظ التعبير العلائقي	التعبير الجبري	التعبير بلغة C++
>	X is greater than Y x أكبر من Y	X > Y	X > Y
<	X is less than Y x أصغر من Y	X < Y	X < Y
>=	X is greater than or equal to Y x أكبر من أو تساوي Y	X ≥ Y	X >= Y
<=	X is less than or equal to Y x أصغر من أو تساوي Y	X ≤ Y	X <= Y
==	X is equal to Y X تساوي Y	X = Y	X == Y
!=	X is not equal to Y X لا تساوي Y	X ≠ Y	X != Y

التعبير العلائقي	ناتج التعبير العلائقي
2 + 5 == 8 – 1	True
5*2 != 10	False
7 % 3 >= 7	False
40 / 5 < 40 / 4	True
20 – 4 * 5 == 0	True
85<= 3 * 20 + 5 * 5	True

التعبيرات الحسابية والعلائقية والمنطقية

❖ مثال :

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int x = 1;
8      int y = 5;
9
10     if (x>y )cout << x << " > " << y << endl;
11     if (x<y )cout << x << " < " << y << endl;
12     if (x>=y )cout << x << " >= " << y << endl;
13     if (x<=y )cout << x << " <= " << y << endl;
14     if (x==y )cout << x << " == " << y << endl;
15     if (x!=y )cout << x << " != " << y << endl;
16
17     return 0;
18 }
```

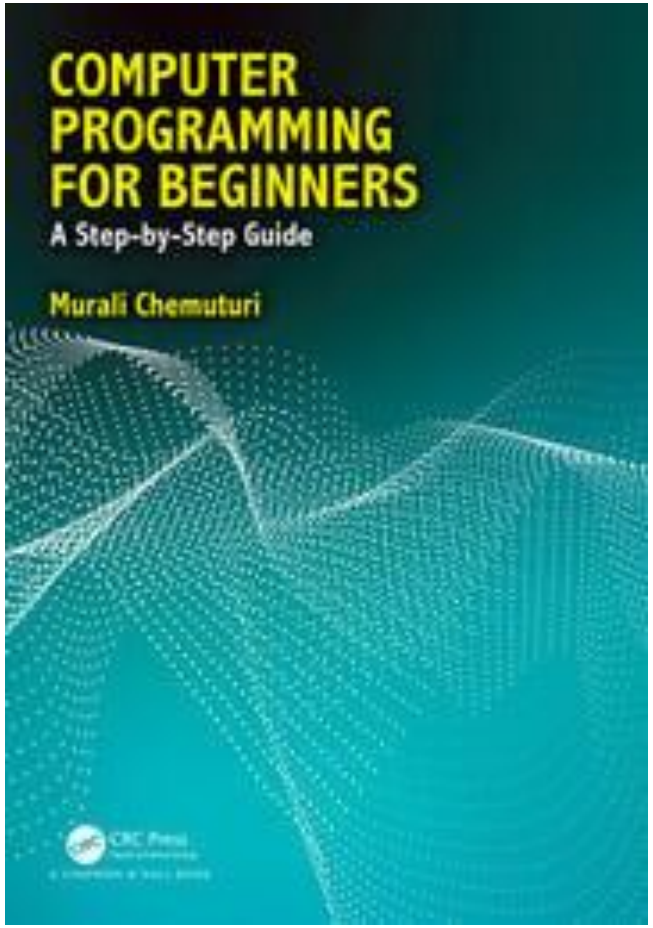
```
1 < 5
1 <= 5
1 != 5

Process returned 0 (0x0)   execution time : 0.022 s
Press any key to continue.
```

روابط خارجية

عنوان الفيديو	الرابط
Cs50 بالعربي	https://youtube.com/playlist?list=PLknwEmKsW8OvMsFbU9zo8oJCprAsgc4LO&feature=shared
ما قبل تعلم البرمجة	https://youtube.com/playlist?list=PLDoPjvoNmBAx8xKvAXpb6f0Urj98Xo7zg&feature=shared

المراجع

- اساسيات برمجة الحاسب الالي
- computer programming for beginners



شكرا لكم