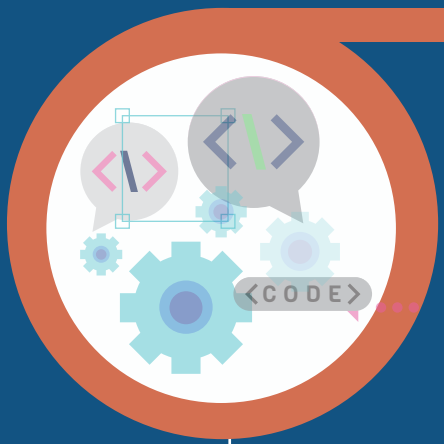
وزارة التربية

تقنية المعلومات

الصف العاشر - الجزء الأول



تقنية المعلومات
الصف العاشر
الجزء الأول
المرحلة الثانوية

تقنية المعلومات

الصف العاشر - الجزء الأول

إشراف

منى سالم عوض سالم (رئيس اللجنة)

علي أحمد الكندري (نائب رئيس اللجنة)

تأليف

رأفت صابر عبدالله أحمد　　حسام الدين علي عبدالقادر

محمد السيد محمد إبراهيم　　أشرف رضوان رضوان سليمان

إبراهيم عبدالله إبراهيم المياس

تصميم

إيمان عبدالعزيز أحمد الفارسي　　سنية محمد علي المؤمن

إخراج

أشرف رضوان رضوان سليمان

الطبعة الأولى

١٤٤٦ هـ

٢٠٢٤ - ٢٠٢٥ م

الطبعة الأولى: ٢٠٢٤-٢٠٢٥ م

حَدَّثَنَا مُحَمَّدُ بْنُ عَبْدِ الأَعْلَى الصَّنْعَانِيُّ حَدَّثَنَا سَلَمَةُ بْنُ رَجَاءٍ حَدَّثَنَا الْوَلِيدُ بْنُ جَمِيلٍ حَدَّثَنَا الْقَاسِمُ أَبُو عَبْدِ الرَّحْمَنِ عَنْ أَبِي أُمَامَةَ الْبَاهِلِيِّ قَالَ ذُكِرَ لِرَسُولِ اللَّهِ -صلى الله عليه وسلم- رَجُلاَنِ أَحَدُهُمَا عَابِدٌ وَالآخَرُ عَالِمٌ فَقَالَ رَسُولُ اللَّهِ -صلى الله عليه وسلم- «فَضْلُ الْعَالِمِ عَلَى الْعَابِدِ كَفَضْلِي عَلَى أَدْنَاكُمْ». ثُمَّ قَالَ رَسُولُ اللَّهِ -صلى الله عليه وسلم- «إِنَّ اللَّهَ وَمَلاَئِكَتَهُ وَأَهْلَ السَّمَوَاتِ وَالأَرْضِ حَتَّى النَّمْلَةَ فِي جُحْرِهَا وَحَتَّى الْحُوتَ لَيُصَلُّونَ عَلَى مُعَلِّمِ النَّاسِ الْخَيْرَ». قَالَ أَبُو عِيسَى هَذَا حَدِيثٌ حَسَنٌ صَحِيحٌ غَرِيبٌ.

☐ الكتاب كاملًا

☐ شاركنا تقييم مناهجنا

طبع في: شركة المطبعة الألمانية للطباعة والتغليف ذ.م.م.

أودع بمكتبة الوزارة تحت رقم (١٨) بتاريخ ٢٤/ ٧/ ٢٠٢٤م

حضرة صاحب السمو الشيخ مشعل الأحمد الجابر الصباح

أمير دولة الكويت

H.H. Sheikh Meshal AL-Ahmad Al-Jaber Al-Sabah
Amir Of The State Of Kuwait

سمو الشيخ صباح خالد الحمد الصباح

ولي عهد دولة الكويت

H. H. Sheikh Sabah Khaled Al-Hamad Al-Sabah
Crown Prince Of The State Of Kuwait

المحـــتوى

المقدمة

في عالم اليوم الذي يعتمد بشكل متزايد على التكنولوجيا، أصبح الأمن السيبراني وبرمجة البايثون من المجالات ذات الأهمية المتزايدة. حيث تعد هذه المجالات من المهارات الأساسية التي يحتاجها الطلاب في حياتهم العملية وتنمية الوطن.

حيث يستخدم الأمن السيبراني لحماية الأنظمة والشبكات والأجهزة الإلكترونية من الهجمات الإلكترونية. التي تُمكن من سرقة البيانات أو إتلافها أو تعطيل الوصول إلى النظام. والتي من شأنها أن يكون لها عواقب وخيمة على الأفراد والشركات والحكومات.

وتعتبر برمجة البايثون لغة برمجة قوية وسهلة التعلم، وهي مفيدة في مجموعة متنوعة من المجالات، بما في ذلك الأمن السيبراني.

ويمكن أن يساعد تعلم الأمن السيبراني وبرمجة البايثون الطلاب في تطوير أنفسهم في حياتهم العملية وتنمية الوطن بعدة طرق، منها:

اكتساب مهارات مهمة في سوق العمل: حيث أصبحت هذه المهارات من المهارات الأساسية المطلوبة في وظائف عدة، خاصة في مجالات التكنولوجيا.

والاستعداد للمستقبل: حيث يتوقع أن تستمر أهمية هذه المجالات في النمو في المستقبل.

المساهمة في تنمية الوطن: حيث يمكن للمتعلمين الاستفادة من هذه المهارات لحماية الوطن من الهجمات الإلكترونية.

كما يهدف هذا الكتاب إلى تزويد طلاب الصف العاشر بالمعرفة والمهارات اللازمة في مجال الأمن السيبراني وبرمجة البايثون.

الوحـــدة الأولـــى

الأمن السيبراني Cyber Security

1 مفاهيم الأمن السيبراني

2 المحاور الرئيسة للأمن السيبراني CIA

3 تطبيقات أمنية للمحاور الرئيسة للأمن السيبراني CIA

4 ممارسات الأمن السيبراني

5 وظائف الأمن السيبراني

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

1- مفاهيم الأمن السيبراني The Concepts of Cyber Security

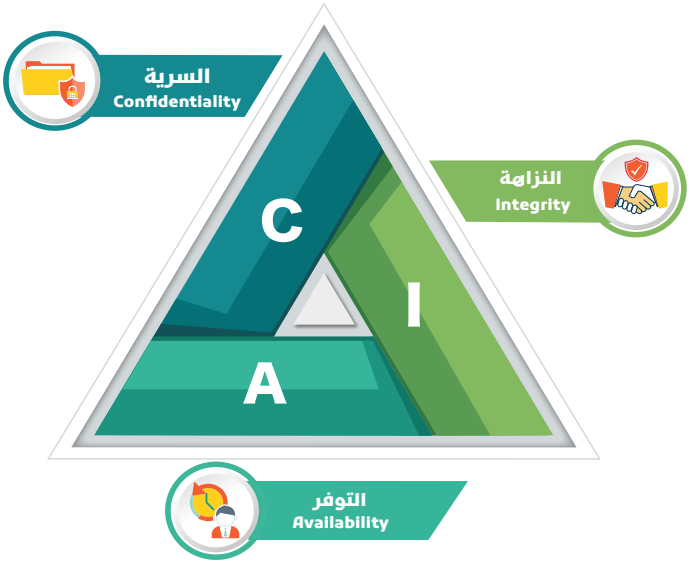
(1-1) أمن المعلومات Information Security

الممارسات والتدابير المتخذة لحماية البيانات والأنظمة من وصول غير المصرح لهم (لاستخدامها - الاطلاع عليها - تعطيلها - تعديلها - تدميرها)، ويشمل حماية المعلومات بشكل عام، بغض النظر عن الوسائط المستخدمة سواءً كانت **رقمية أو غير رقمية.**

(1-2) الأمن السيبراني Cyber Security

يركز على حماية الأنظمة والشبكات من التهديدات الإلكترونية، وحماية البيانات **الرقمية** والبرامج من الهجمات _التي تهدف للوصول إلى المعلومات المهمة أو تغييرها أو تدميرها، ويتم تطبيق الإجراءات، والضوابط، والعمليات، والتقنيات لتقليل مخاطر الهجمات السيبرانية والحماية من الاستغلال غير المصرح للأنظمة والشبكات والتقنيات.

2- المحاور الرئيسة لأمن المعلومات والأمن السيبراني CIA



يعد نموذج CIA أداة أساسية لضمان أمن المعلومات. من خلال التركيز على السرية والنزاهة والتوفر، يمكن للمؤسسات حماية بياناتها من الوصول غير المصرح به، وضمان دقتها واكتمالها، وجعلها متاحة للمستخدمين المصرح لهم عند الحاجة، ومن الآثار المترتبة على انتهاكها (انتهاك الخصوصية - سرقة الهوية - الإضرار بالسمعة - الخسارة المالية - عدم التمكن من الوصول للخدمات الأساسية والطوارئ والتواصل).

(2-1) السرية Confidentiality

التأكد من حماية المعلومات التي لا يمكن الوصول إليها والحفاظ على سريتها إلا من قبل الأفراد المصرح لهم فقط Authorized Individuals. مثل (الصور الخاصة، والبيانات المصرفية، وبيانات الدراسة أو العمل، والوثائق الحكومية).

(2-2) النزاهة Integrity

تشير النزاهة إلى الحفاظ على صحة، وموثوقية المعلومات الرقمية (Authenticity, Reliability)، وسلامتها. لضمان عدم قيام أي أطراف غير مصرح لها Unauthorized بالتلاعب بالمعلومات التي ترسلها وتستقبلها عبر الإنترنت.

(2-3) التوفر Availability

يقصد بالتوفر التأكد من أن المعلومات والموارد متاحة باستمرار Consistently Available، وقابلة للاستخدام usable عند الحاجة. وإمكانية الوصول إلى الأنظمة والخدمات وتشغيلها عند الحاجة، وحماية أنظمة المعلومات وبياناتها من الانقطاعات أو التوقف عن العمل.

أمثلة على الهجمات التي تؤثر على المحاور الرئيسية للأمن السيبراني

CIA TRIAD

Confidentiality السرية

Password cracking

استخدام هجمات مختلفة (خوارزميات وتطبيقات) لتخمين كلمة المرور أو كشفها والوصول إلى حساب المستخدم.

Dumpster diving

يحصل المخترق على المستندات أو البيانات الحساسة التي لم يتم التخلص منها نهائيًا بطريقة آمنة، لشن هجوم سيبراني.

Phishing

محاولة سرقة معلومات حساسة، عادةً ما تكون في شكل أسماء مستخدمين أو كلمات مرور أو أرقام بطاقات ائتمان أو معلومات حساب مصرفي أو بيانات مهمة أخرى.

Integrity النزاهة

Man In The Middle (MITM) attack

هجوم إلكتروني حيث يقوم المخترق بنقل وتغيير الاتصالات بين طرفين يعتقدان أنهما يتواصلان مباشرة مع بعضهما البعض، وأنهما يجريان تبادل للمعلو ما ت .

Availability التوفر

DoS* (Denial of Service)& DDoS* (Distributed Denial of Service) Attacks

نوع من الهجمات السيبرانية الذي يستهدف مواقع الإنترنت والخوادم (Servers)، حيث يتم إغراقها بعدد هائل من الطلبات وحركات المرور* (Traffic)، مما يؤدي إلى إضعاف خدمات موقع الإنترنت أو تعطيله تماماً.

* هجمات DoS: تستهدف جهازًا أو خدمة واحدة بغرض إغراقها بطلبات زائفة، ومن السهل تتبع هجماتها.

* هجمات DDoS: تُنفذ من خلال أجهزة متعددة ترسل حزمًا من البيانات، وتهاجم من مواقع متعددة. مما يزيد من سرعتها، وقوة الهجوم وصعوبة التصدي لها، ومن الصعب تتبع هجماتها.

* حركة مرور الويب (Traffic): هي كمية البيانات التي يتم إرسالها واستقبالها من قبل زوار ومستخدمي موقع الإنترنت.

تذكر أن تطبيق مبادئ CIA في الأمن السيبراني مسؤولية مشتركة. من خلال ممارسة هذه الخطوات البسيطة وتعزيز الوعي، يمكننا إنشاء بيئة رقمية أكثر أمانًا وموثوقية للجميع.

ويجب أن ننتبه أن مبادئ CIA هي أهم ما يحدد آلية تأمين المؤسسات والأفراد في مجال الأمن السيبراني، ويتم تحديد أولويات تنفيذ السياسات الأمنية للمبادئ الثلاثة (السرية، النزاهة، التوفر) وفقًا لعملية إدارة المخاطر بالمؤسسة.

مثال (2-1): جهاز الصراف الآلي ATM Automated Teller Machine:

- **(السرية Confidentiality):**

تطبيق المصادقة الثنائية مثل: (البطاقة البنكية Bank Card، ورمز المرور PIN)، للسماح للمستخدم الوصول للبيانات المطلوبة، حفاظًا على سرية المعلومات المصرفية.

- **(النزاهة Integrity):**

ضمان سلامة البيانات حيث لا يمكن لأي شخص تغيير معلومات الحسابات المصرفية ما لم يكن مصرحًا له، من خلال إبلاغ المستخدم عن أي عمليات مصرفية تمت عبر جهاز الصراف الآلي.

- **(التوفر Availability):**

النظام المصرفي الإلكتروني (جهاز الصراف الآلي ATM، الموقع Website، والتطبيق Application) متاح في أي وقت إمكانية الاستخدام في الأماكن العامة، والوصول إليه على مدار الساعة.

3- تطبيقات أمنية للمحاور الرئيسية للأمن السيبراني CIA

(1-3) المصادقة Authentication

المصادقة هي طبقة إضافية أو أكثر من الأمان، تستخدم لحماية حسابات المستخدمين. بتقديم عدة أشكال منفصلة لتحديد الهوية قبل منح الوصول. وتتمثل في:

1- عامل المعرفة: (ما تعرفه Some thing you **Know**) وهو شيء يعرفه المستخدم.

2- عامل الامتلاك: (ما تملكه Some thing you **Have**) وهو شيء يمتلكه المستخدم.

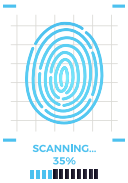
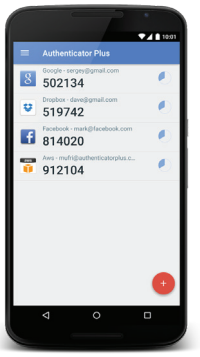
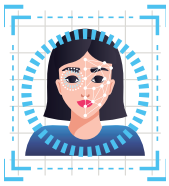
3- عامل المقياس الحيوي: (ما أنت عليه Some thing you **Are**) هو الأكثر ملاءمةً وإقناعاً لإثبات هُوِية المستخدم.

ومن أمثلة عامل المقياس الحيوي، المصادقة البيومترية **Biometric Authentication** تستخدم خصائص بيولوجية فريدة، مثل بصمات الأصابع أو مسح قزحية العين أو التعرف على الوجه، للتحقق من هوية المستخدم. تضيف هذه الطريقة طبقة إضافية من الأمان حيث يصعب تكرار هذه السمات المادية أو تزويرها.

أصبحت المصادقة البيومترية شائعة بشكل متزايد في الهواتف الذكية وأجهزة الكمبيوتر المحمولة، مما يوفر طريقة آمنة لمصادقة دخول المستخدمين وحماية معلوماتهم الشخصية.

وعلى سبيل المثال استخدام المصادقة البيومترية (التعرف على الوجه) في تطبيق هويتي.

جدول يوضح بعض عوامل المصادقة المستخدمة

1- عامل المعرفة (ما تعرفه) Something you Know	2- عامل الامتلاك (ما تملكه) Something you Have	3- عامل المقياس الحيوي (ما أنت عليه) Something you Are
كلمات المرور Passwords 	Hardware token 	بصمة العين 
مصادقة الرسائل القصيرة OTP -PIN 	Smart card 	بصمة اليد 
المصادقة المستندة إلى التطبيق 	Smart phone 	بصمة الوجه 

(2-3) الشبكة الافتراضية الخاصة Virtual Private Network (VPN)

تعد الشبكة الافتراضية الخاصة (VPN) أداة مهمة لضمان الخصوصية عبر الإنترنت. من خلال تشفير اتصال الإنترنت وتوجيهه عبر خادم Server موجود في موقع Location مختلف.

تنشئ شبكات VPN اتصالًا آمنا يخفي عنوان IP الخاص بالمستخدم ونشاط التصفح، يساعد هذا في حماية البيانات من المهاجمين وحماية الخصوصية، خاصة عند استخدام شبكات Wi-Fi العامة.

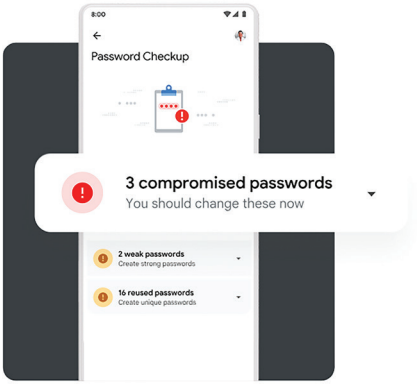
(3-3) تطبيقات المراسلة الآمنة (التشفير) Secure Messaging Apps

توفر تطبيقات المراسلة الآمنة تشفيرًا من طرف إلى طرف لحماية محادثات المستخدمين والتأكد من أن المستلمين المقصودين فقط يمكنهم الوصول إلى الرسائل. تمنع هذه التطبيقات أي طرف ثالث، بما في ذلك مزودي الخدمة والمتسللين، من اعتراض الرسائل أو قراءتها، يمكن التواصل بسرية دون المساس بالخصوصية.

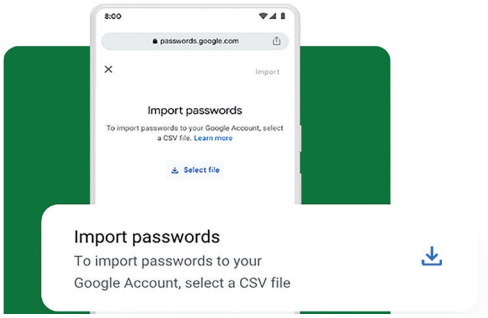
(4-3) برامج إدارة كلمات المرور Password Managers

تعتبر برامج إدارة كلمات المرور من الأدوات الرقمية التي تساعد على تقديم الحماية والأمان لحسابات المستخدمين، وذلك من خلال المميزات الآتية:

- إنشاء كلمات مرور قوية وفريدة، وتخزينها وإدارتها بصورة آمنة.
- اكتشاف كلمات المرور الضعيفة لتقليل خطر التعرض للاختراق.



- الوصول إليها تلقائيًا عند الحاجة، ودعم خاصية المزامنة بين الأجهزة.



- إنشاء رموز المصادقة الثنائية، ودعم خاصية مفاتيح الأمان U2F(Universal 2nd Factor).
- دعم إدارة بطاقات الائتمان عبر الإنترنت، مع طبقة إضافية من الأمان.

ومن أشهر تطبيقات إدارة كلمات المرور (Lastpass - 1password - Google Password Manager).

(5-3) تشفير البريد الإلكتروني Encryption for Email

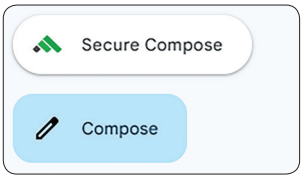
توفير طبقة إضافية من الحماية للمعلومات التي تتم مشاركتها عبر البريد الإلكتروني. حيث يتم تشفيرها بطريقة لا يمكن إلا للمُرسل إليه فك تشفيرها وقراءتها.

وهناك العديد من الخدمات الرقمية مثل Pretty Good Privacy (PGP) أو Secure S/MIME) Multipurpose Internet Mail Extension)، التي تتيح تشفير الرسائل الإلكترونية، سنتناول منها أداة FlowCrypt:

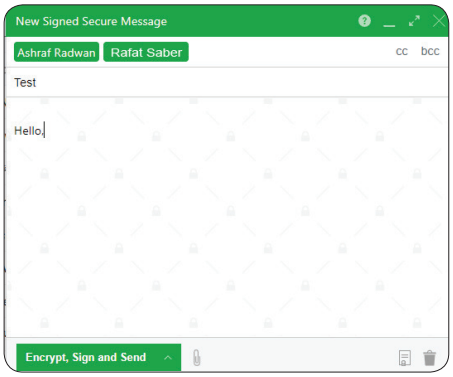
- تثبيت الأداة من موقع Chrome Web Store الخاص بمتصفح Chrome.

- إنشاء حساب جديد، ومنحه الصلاحيات اللازمة لإدارة عملية تشفير البريد الإلكتروني Gmail.

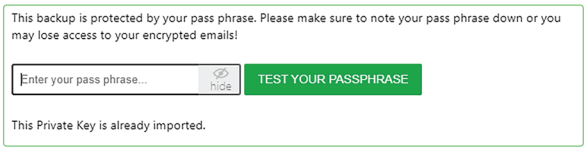
- اختيار إرسال رسالة إلكترونية جديدة مشفرة.



- تحديد جهات الاتصال، ثم كتابة عنوان ومحتوى الرسالة، وإرسالها.



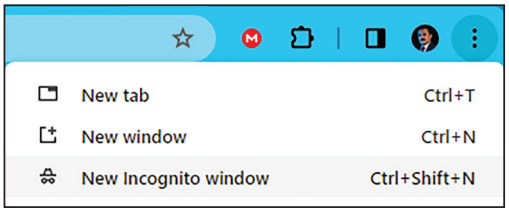
- يفتح المستقبل الرسالة بعد إدخال رمز التشفير الصحيح.

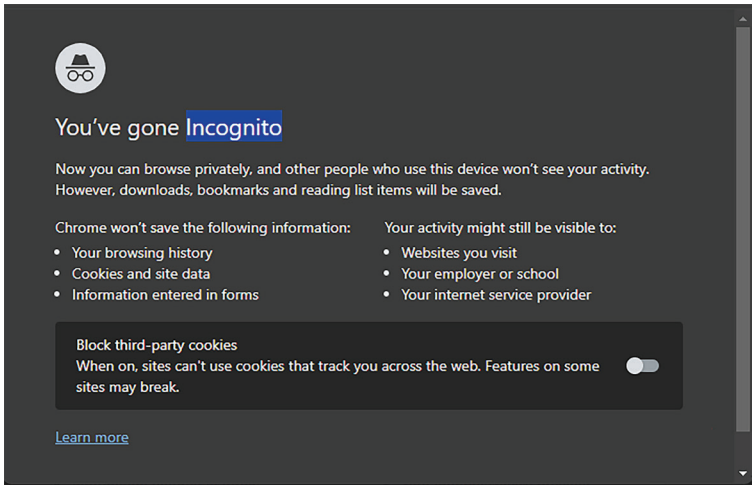


(6-3) الخصوصية في متصفحات الإنترنت Privacy-Focused Web Browsers

توفر متصفحات الويب الآمنة مثل (Mozilla Firefox أو Brave) تعزيز ميزات الأمان. عن طريق التحكم بإعدادات وإضافات تتيح للمستخدمين الحفاظ على مستوى أعلى من الخصوصية أثناء تصفح الإنترنت، ومنها خاصية التصفح المتخفي Incognito بمتصفح Google Chrome، حيث توفر الخصوصية لأي مستخدم لمتصفح الإنترنت من خلال:

- عدم الاحتفاظ بسجل التصفح الخاص بك.
- عدم الاحتفاظ بملفات تعريف الارتباط وبيانات الموقع.
- عدم الاحتفاظ بالمعلومات المدخلة في النماذج (اسم المستخدم وكلمات المرور).





(7-3) المحافظ الرقمية وطرق الدفع الآمنة Digital Wallets and Secure Payment Methods

توفر المحافظ الرقمية مثل (Apple Pay أو Google Wallet) طرق دفع آمنة لحماية المعلومات المالية، والحفاظ على المعلومات وعدم استغلالها، وتقليل مخاطر الوصول إليها.

ورقة عمل 1 (ورقة عمل لا صفية)

إنشاء مصادقة باستخدام تطبيق Google Authenticator:

1- حمل التطبيق من متاجر الهواتف الذكية.

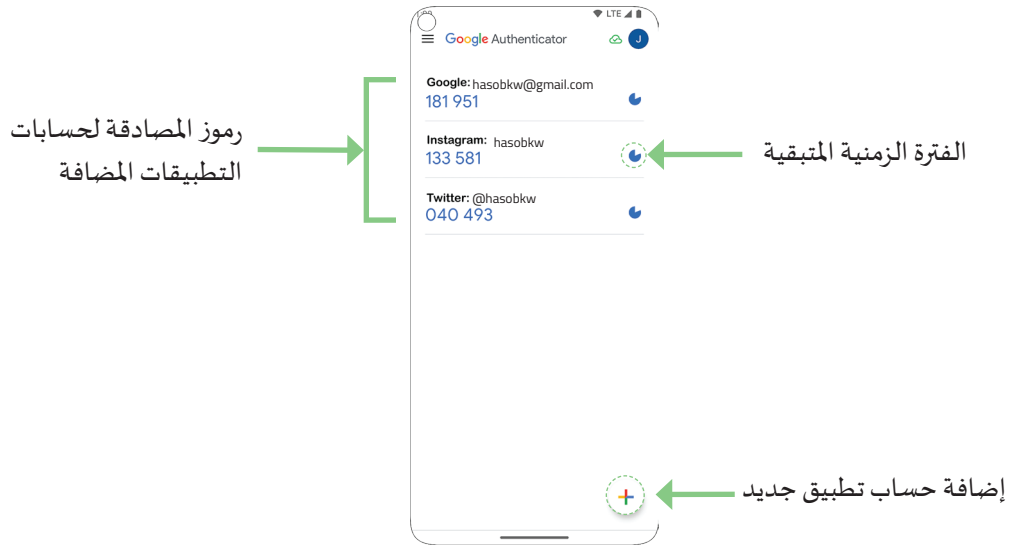
Apple store Anriod Store

2- أنشئ حساب على التطبيق من خلال حساب Google.

3- أضف التطبيقات المختلفة المراد تأمينها من خلال المصادقة، مثال على ذلك تطبيق التواصل الاجتماعي Instagram.

4- أدخل Login على تطبيق Instgram، بإدخال كلمة المرور الخاصة بالتطبيق.

5- سيطلب منك تطبيق Instgram إدخال الرمز الذي يتم توليده باستخدام تطبيق المصادقة Google Authenticator، كما هو موضح بالشكل.



6- نسخ الرمز ثم إضافته في تطبيق Instagram، خلال الفترة الزمنية المحددة.

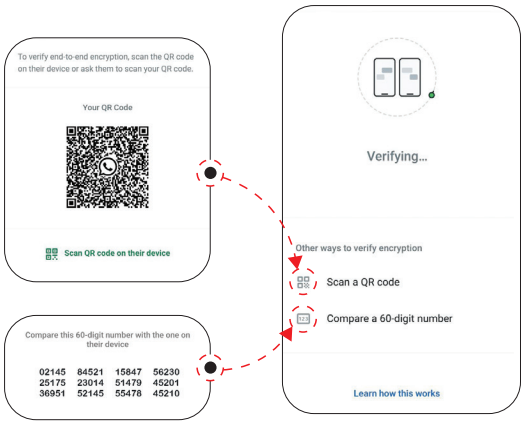
7- تشغيل تطبيق Instagram بشكل آمن.

ورقة عمل 2 (ورقة عمل لا صفية)

تطبيق WhatsApp: حيث تتميز المحادثات المشفرة بينك وبين شخص آخر برمز أمان خاص بها. يُستخدم هذا الرمز للتحقق من أن المكالمات والرسائل التي ترسلها إلى تلك الدردشة مشفرة تمامًا.

ويمكن الوصول لهذا الرمز من خلال:

- شاشة معلومات* جهة الاتصال Contact Info الخاصة بأحد الأشخاص الذين تتواصل معهم.
- اختيار Encryption.
- التحقق من رمز الأمان Verify encryption.
- اختيار رمز من رمز QR أو الرقم المكون من 60 رقماً.



تُعد هذه الرموز فريدة لكل محادثة، ويمكن مقارنتها مع الطرف الآخر للتحقق من أن الرسائل بين الطرفين مشفرة.

ملاحظة:

* يجب مراعاة اختلافات الشاشات في أنظمة تشغيل الهواتف الذكية.

يعتبر التحقق من رمز الأمان عملية اختيارية للمحادثات المشفرة بين طرفين.

ورقة عمل 3 (ورقة عمل لا صفية)

- ناقش مع معلمك وأسرتك كيفية إجراء المصادقة الثنائية بين تطبيق سهل وتطبيق هويتي، ثم سجل الخطوات هنا.

4- ممارسات الأمن السيبراني Cybersecurity Practices:

يمكن اتخاذ بعض الإجراءات والتدابير الخاصة من أجل الحفاظ على CIA في الأمن السيبراني:

- **كلمات مرور قوية Strong passwords:** استخدم كلمات مرور قوية وفريدة لكل الحسابات عبر الإنترنت، وتجنب كلمات المرور التي يمكن تخمينها.
- **المصادقة Authentication:** تفعيل المصادقة الثنائية أو المتعددة لحساباتك المختلفة حال توافرها.
- **حماية الأجهزة Security:** تثبيت برامج مكافحة الفيروسات والبرامج الضارة، وتحديث البرامج بانتظام واستخدام تشفير Encryption قوي للبيانات المهمة.
- **مكافحة التصيد الاحتيالي Phishing:** عدم فتح روابط Links لا تعرف مصدرها أو تحميل مرفقات من مصادر غير معروفة، وعدم الوقوع فريسة لرسائل البريد الإلكتروني أو الرسائل التي تحاول استدراجك للكشف عن معلومات شخصية.
- **استخدام مواقع الويب والتطبيقات الموثوقة:** التزم بالأنظمة الأساسية ذات الإجراءات الأمنية القوية والمصادر التي لديها موثوقية.
- **الحذر على وسائل التواصل الاجتماعي:** تقليل المعلومات الشخصية التي تشاركها عبر الإنترنت، وتفعيل إعدادات الخصوصية لكل تطبيق.
- **إدارة عملية مشاركة البيانات Data sharing:** التعرف على كيفية مشاركة البيانات عبر الإنترنت، وضبط إعدادات الخصوصية وفقًا لذلك.
- **الإبلاغ عن أي نشاط مشبوه Report:** الإبلاغ عن أي نشاط مشبوه أو انتهاكات مشتبه بها إلى مسئولي النظام الأساسي، أو السلطات المختصة (إدارة مكافحة الجرائم الإلكترونية*).
- **التحقق من مصادر المعلومات:** تأكد من صحة المعلومات ومصدرها قبل الوثوق بها.
- **النسخ الاحتياطي للبيانات Data backup:** يجب التأكد من إجراء عمليات النسخ الاحتياطي للملفات والمستندات المهمة بانتظام لضمان الوصول إليها في حال تم اختراق النسخة الأصلية منها.

* إدارة مكافحة الجرائم الإلكترونية تتلقى جميع البلاغات بخصوص كل ما يخالف القانون، وتقوم بإجراء التحريات للتأكد من صحة البلاغات وجدية المعلومات واتخاذ الإجراءات القانونية اللازمة حيالها. للبلاغات يرجى الاتصال على رقم الطوارئ 97283939، سيتم التعامل معك بسرية تامة.

5- الوظائف في مجال الأمن السيبراني

يقدم مجال الأمن السيبراني مجموعة واسعة من الوظائف، ويركز كل منها على جوانب مختلفة لحماية الأنظمة والمعلومات الرقمية من الوصول غير المصرح به ومواجهة الهجمات والتهديدات، وفيما يلي بعض الوظائف الشائعة للأمن السيبراني:

1- محلل الأمن السيبراني Cybersecurity Analyst :

مراقبة أنظمة الكمبيوتر والشبكات والتطبيقات لاكتشاف انتهاكات الأمان أو الأنشطة المشبوهة. وتحليل مخاطر الأمان، والتحقيق في الحوادث السيبرانية، وتنفيذ تدابير لمنع الهجمات المستقبلية.

2- مهندس أمن Security Engineer:

تصميم وتطوير أنظمة الأمان وتنفيذها لحماية الشبكات والبنية التحتية للأنظمة، (الجدران النارية وأساليب التشفير وأنظمة الكشف عن الاختراق). وكذلك إجراء تقييم للثغرات باختبارها لتحديد ومعالجة مواطن الضعف في الأنظمة.

3- مستشار أمني Security Consultant:

تقديم نصائح وإرشادات متخصصة للمؤسسات حول كيفية تحسين وضعها الأمني. وتقييم المخاطر، وتطوير سياسات الأمان والإجراءات، وإنشاء خطط استجابة للحوادث، وتقديم توصيات لتعزيز ضوابط الأمان.

4- مخترق أخلاقي Ethical Hacker:

تحديد الثغرات في أنظمة الكمبيوتر والشبكات، والقيام بمحاولات اختراق مصرح بها لمحاكاة الهجمات الحقيقية ومساعدة المؤسسات في تعزيز دفاعاتها. يحتاج القراصنة الأخلاقيون إلى فهم عميق لتقنيات القرصنة ولغات البرمجة وأدوات الأمان.

5- مهندس تصميم أنظمة أمنية Security Architect:

تصميم أنظمة وشبكات تكنولوجيا المعلومات الآمنة، وتطوير أطر الأمان وإنشاء سياسات الأمان، وضمان التطبيق للمعايير والتشريعات الصناعية. يتعاون مهندسو بنية الأمان مع أصحاب المصلحة المختلفين لمواءمة متطلبات الأمان مع أهداف العمل التجارية.

6- المستجيب للحوادث Incident Responder :

التحقيق في الحوادث، والاستجابة للحوادث الأمنية، مثل اختراقات البيانات أو العدوى بالبرامج الضارة، وتحليل أنماط الحوادث، واحتواء التهديدات، وتحليل مدعوم بالأدلة لتحديد سبب الحوادث. كذلك تطوير خطط استجابة للحوادث، وتقديم توصيات لتحسين قدرات كشف واستجابة الحوادث.

7- محلل مركز العمليات الأمنية Security Operations Center (SOC) Analyst:

مراقبة الأحداث والتنبيهات الأمنية، والاستجابة للحوادث الأمنية المحتملة. من خلال استخدام أدوات إدارة المعلومات والأحداث الأمنية (MEIS) لتحديد التهديدات، وتحليل بيانات السجل، واتخاذ الإجراءات المناسبة. كذلك يلزم محللو SOC إلى معرفة تقنيات الأمان وإجراءات التعامل مع الحوادث.

الوحدة الثانية: برمجة Python

مدخل إلى البرمجة

Introduction to programming

نواتج التعلم

- التعرف على مفهوم البرمجة.
- تحديد مراحل إنشاء برنامج حاسوبي.
- كتابة خوارزمية لحل مشكلة ما.
- تصميم خريطة تدفق لإجراءات حل مشكلة.
- التعرف على لغة بايثون.
- استخدام محرر الأوامر PyCharm.
- التعرف على الدالة المدمجة print().
- التعرف على هياكل البيانات.
- معرفة أنواع هياكل البيانات في بايثون.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

مدخل إلى البرمجة Introduction to programming

1- مقدمة

تُعد البرمجة مهارة أساسية تُمكنك من تحقيق العديد من الفوائد، مثل:

- حل المشكلات:

يمكنك استخدام البرمجة لإنشاء برامج تُحل مشكلات محددة في حياتك أو عملك، مما يُحسّن من كفاءتك وإنتاجيتك.

- الإبداع:

تُتيح لك البرمجة فرصة الإبداع من خلال إنشاء ألعاب، وتطبيقات، ومواقع إلكترونية، وبرامج أخرى تُعبّر عن أفكارك ورؤيتك.

- تحسين مهاراتك:

يُساعدك تعلم البرمجة على تحسين مهارات حل المشكلات، والتفكير المنطقي، والإبداع، وهي مهارات أساسية للنجاح في أي مجال.

- توفير فرص وظيفية جديدة:

يُمكنك تعلم البرمجة للحصول على وظيفة في مجال التكنولوجيا، مثل تطوير البرامج، تحليل البيانات، أو الذكاء الاصطناعي، وهي مجالات تُعاني من نقص في الكوادر المؤهلة.

(1-1) مفهوم البرمجة

هي مجموعة من التعليمات والأوامر تترجم للغة الآلة توجه الحاسوب لتنفيذ مجموعة من القواعد، والتعليمات لحل مشكلة ما.

(1-2) خطوات إنشاء برنامج

تمر عملية إنشاء برنامج حاسوبي بمجموعة من الخطوات والإجراءات.

أ- تحديد المشكلة

تُعرف المشكلة على أنها موقف حياتي يتطلب حلول محددة للوصول لهدف ما.

فمثلًا إذا كنا نرغب في إيجاد مساحة المستطيل، فأول سؤال يتبادر للذهن، ماهي (المعطيات) المتاحة؟، حيث نحتاج لمعرفة طول وعرض المستطيل، بعد ذلك نحتاج لمعرفة العمليات والإجراءات اللازمة للوصول إلى الحل، حيث سيكون القانون، المساحة = الطول × العرض الذي من خلاله سيتم إيجاد الناتج النهائي المطلوب.

ب- إعداد خطة الحل (الخوارزمية Algorithm)

الخوارزمية عبارة عن مجموعة من الخطوات المنطقية والمتسلسلة لحل مشكلة ما، وقد أُطلق عليها هذا الاسم نسبةً إلى محمد بن موسى الخوارزمي، عالم الرياضيات المسلم، وهي طريقة وصفية توضح خطوات حل المشكلة باستخدام المعطيات (المدخلات) للوصول إلى الناتج أو الهدف (المخرجات).

يمثل التخطيط التالي (خوارزمية) حساب مساحة المستطيل:

- تحديد قانون حساب مساحة المستطيل.
- تحديد قياس الطول والعرض.
- تطبيق القانون لحساب المساحة.

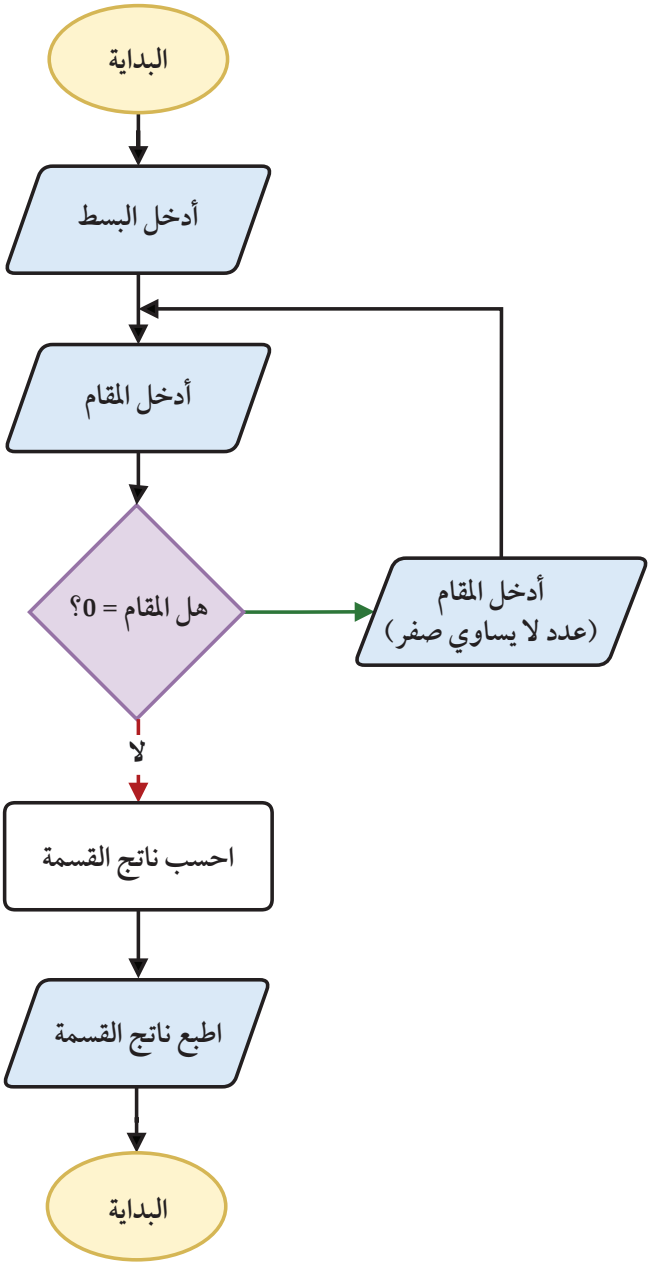
ج- خرائط التدفق (Flowchart)

إحدى أنواع المخططات الرسومية التي توضح ترتيب العمليات اللازمة لحل مشكلة ما، حيث تستخدم أشكالًا قياسية متفق عليها للدلالة على العمليات المختلفة، والتي تُسهل على المبرمج فهمها لتحويلها إلى برنامج مكتوب بإحدى لغات البرمجة، والجدول رقم (1-1) يوضح وظيفة كل شكل:

الشكل	الوصف والوظيفة
	بداية أو نهاية البرنامج End / Start
	عمليات الإدخال أو الإخراج Input / Output
	عملية المعالجة Process
	اتخاذ قرار (التفرع) Decision
	خطوط الاتجاه والتوصيل بين الأشكال Arrows
	الواصلة في نفس الصفحة On-page connector
	الواصلة بين الصفحات Off-page connector

جدول (1-1) أشكال ووظائف المخططات الرسومية

فإذا أردنا التخطيط لعمل برنامج يُمكن المستخدم من حساب ناتج القسمة لرقمين، مع الأخذ بالاعتبار أنه لا يمكن القسمة على (صفر)، فستكون خريطة التدفق شكل (1-1) كالتالي:



شكل (1-1) خريطة التدفق / لإيجاد ناتج القسمة

د- كتابة البرنامج

يتم تحويل خريطة التدفق لمجموعة من التعليمات بإحدى لغات البرمجة، حيث تختلف لغات البرمجة في قواعد كتابة أوامرها واستخداماتها.

هـ- اختبار البرنامج

يتم تشغيل البرنامج وإدخال بيانات ومقارنة النتائج لاختبار صحته وعمله بصورة سليمة، لتفادي ظهور بعض الأخطاء أثناء كتابة البرنامج أو أثناء التنفيذ.

و- بناء البرنامج

يتم بناء نسخة تنفيذية من البرنامج أو التطبيق أو نشره على الإنترنت.

ز- توثيق البرنامج

يتم توثيق البرنامج بكتابة نوع البيانات المطلوبة للإدخال، وخطوات الحل (الخوارزمية) وكذلك خريطة التدفق، ولغة البرمجة المكتوب بها البرنامج وآخر تعديل، ليتمكن أي مبرمج آخر من تطوير البرنامج.

(3-1) لغة البرمجة Python

تعتبر لغة Python من لغات البرمجة عالية المستوى، مفتوحة المصدر، قابلة للتوسع، سهلة التعلم وقوية الاستخدام. تُستخدم على نطاق واسع في مختلف المجالات منها: تطوير تطبيقات الويب، برامج سطح المكتب، أدوات تحكم الأتمتة (مجموعة من الإجراءات لإنجاز مهمة ما بأقل قدر من التدخل البشري)، تطبيقات الشبكة، علم البيانات، والتعلم الآلي، تحليل البيانات الضخمة، وإنشاء نماذج تنبؤية، وذلك لسهولة استخدامها ووجود العديد من المكتبات المتخصصة. كذلك يتمتع مجتمع بايثون بدعم كبير من المطورين والمبرمجين، مما يسهل الحصول على المساعدة والدعم المستمر.

تمتاز لغة Python بالعديد من المميزات التي جعلتها شائعة الاستخدام وواسعة الانتشار منها:

- سهولة التعلم والاستخدام، مما يجعلها اللغة المفضلة لتعلم البرمجة.
- يمكن تطوير برمجيات معقدة باستخدمها.
- تستخدم في مجالات عدة مثل: تطبيقات الويب، والذكاء الاصطناعي، وتحليل البيانات، وتطبيقات الشبكات.
- تحتوي على العديد من المكتبات الجاهزة.
- تتوافق مع العديد من أنظمة التشغيل.
- يتم تحديثها بصورة مستمرة.

(4-1) الفرق بين Compiler و Interpreter

تعتمد لغات البرمجة على مبدأ تحويل الأوامر البرمجية من تمثيلٍ عالي المستوى إلى تعليماتٍ تنفيذية بلغة الآلة يستطيع المعالج فهمها، والتعامل معها وتنفيذها. من خلال نظام معالجات لغات البرمجة Programming languages processing system، والذي يشير إلى الخطوات المطلوب تنفيذها من أجل تحويل الأوامر البرمجية إلى لغة الآلة، ويتم عبر برامجٍ وسيطة هي المترجم Compiler والمفسر Interpreter، ومن أهم ما يميزهما ما يلي:

Interpreter:

- يقوم بترجمة سطر سطر أو فقرة من البرنامج.
- يستمر في ترجمة وتنفيذ البرنامج وإذا كان هناك خطأ يتوقف في السطر الذي يحتوي على الخطأ أثناء عملية التنفيذ.
- يقوم بترجمة البرنامج بسرعة لكن ينفذه ببطئ.
- سهولة تتبع الأخطاء لأن البرنامج يتوقف في مكان الخطأ مباشرة.

Compiler:

- يقوم بتحليل البرنامج بالكامل ثم يترجمه كاملا إلى لغة الآلة.
- يأخذ وقت طويل في ترجمة البرنامج لكن ينفذه بسرعة كبيرة.
- يكتشف الأخطاء بعد تحليل البرنامج بالكامل لهذا تحديد مكان الخطأ صعب بالمقارنة مع interpreter.

(1-5) كتابة البرامج في بايثونPython

- يمكن استخدام تطبيقات معالجات النصوص مثل Notepad، لكتابة التعليمات البرمجية بلغة Python ومن ثم حفظ الملف بامتداد py.
- يمكن استخدام إحدى بيئات التطوير المتكاملة Integrated Development Environment (IDE) التي توفر الأدوات اللازمة لكتابة التعليمات وتحريرها واختبارها وتصحيح أخطائها في مكان واحد مثل:
  - العديد من مواقع تحرير التعليمات البرمجية على الإنترنت مثل:

    onlinegdb.com و replit.com و online-python.com
  - Visual Studio Code: الخاص بشركة Microsoft.
  - محرر PyCharm: الذي سيتم استخدامه خلال هذا الكتاب.

(1-6) بيئة التطوير المتكاملة PyCharm

تحميل PyCharm

يمكن الحصول على آخر إصدار من محرر PyCharm من خلال الموقع الرسمي على الرابط:

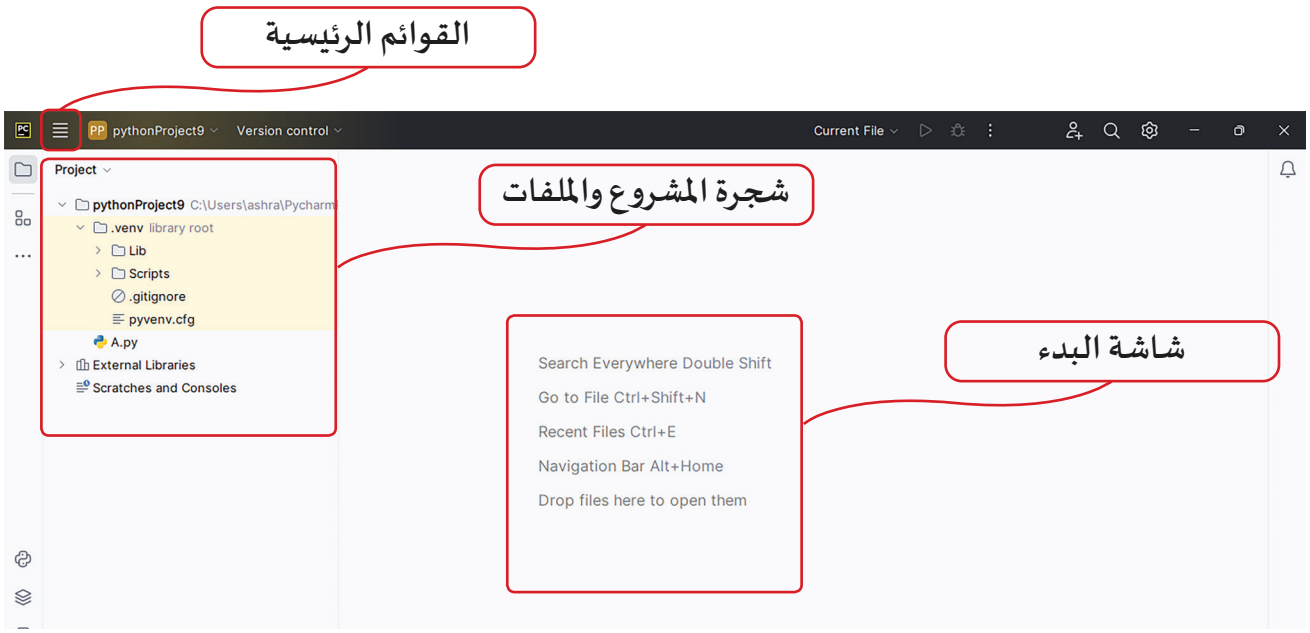
والذي يوفر نسختين:

الأولى: PyCharm Professional وهي نسخة غير مجانية؛ متقدمة للمحترفين.

الثانية: PyCharm Community Edition وهي نسخة مجانية؛ تستخدم للتعليم.

تحميل برنامج PyCharm

(7-1) نافذة المحرر

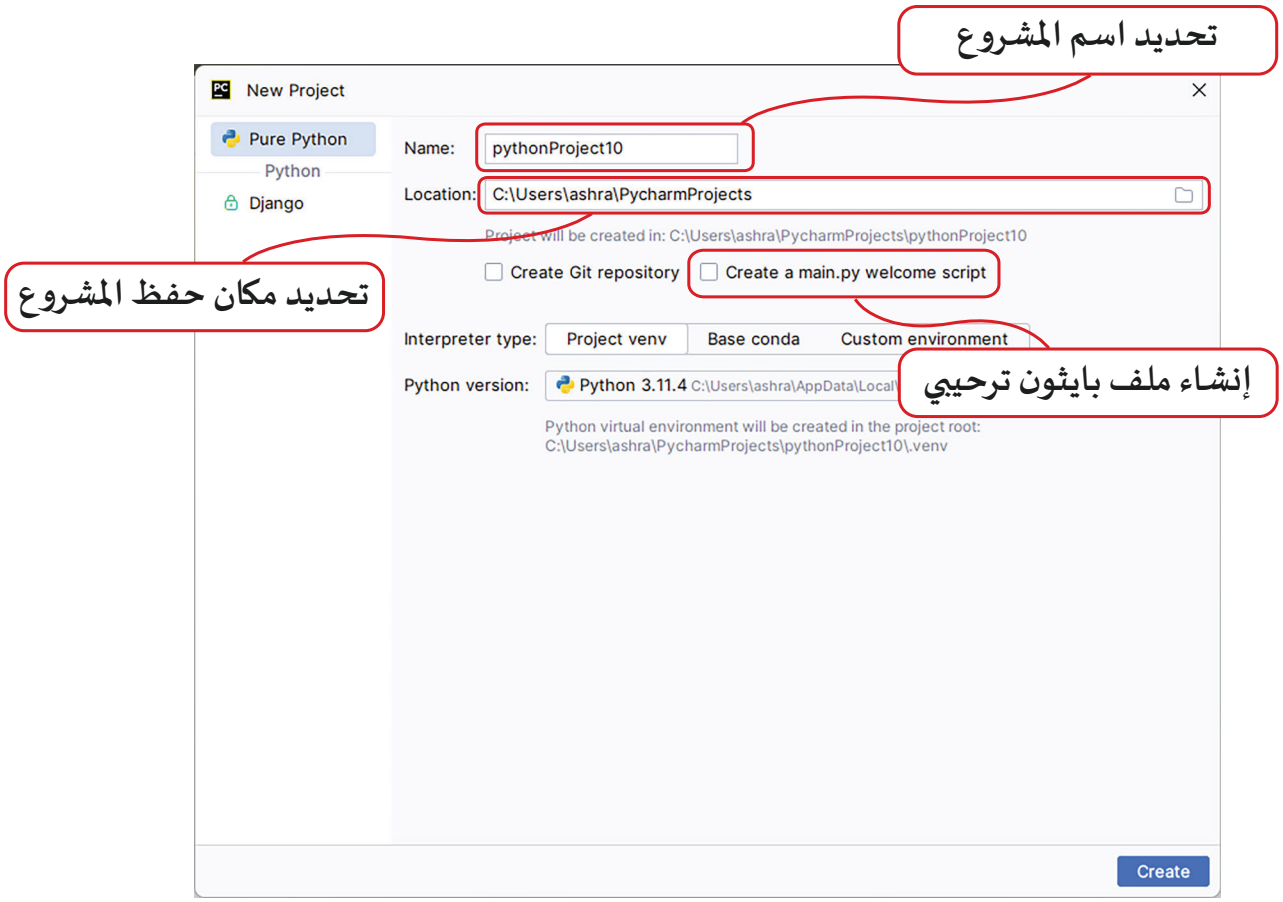


شكل (1-2) شاشة PyCharm Editor

- **القوائم الرئيسية Main Menu:** لإظهار شريط قوائم البرنامج.
- **شجرة المشروع والملفات Project:** يظهر مسار المشروع الحالي وملفات بايثون التي يحتويها.
- **شاشة البدء Welcome Screen:** تظهر مفاتيح روابط لبعض العمليات الشائعة.

(8-1) إنشاء مشروع جديد

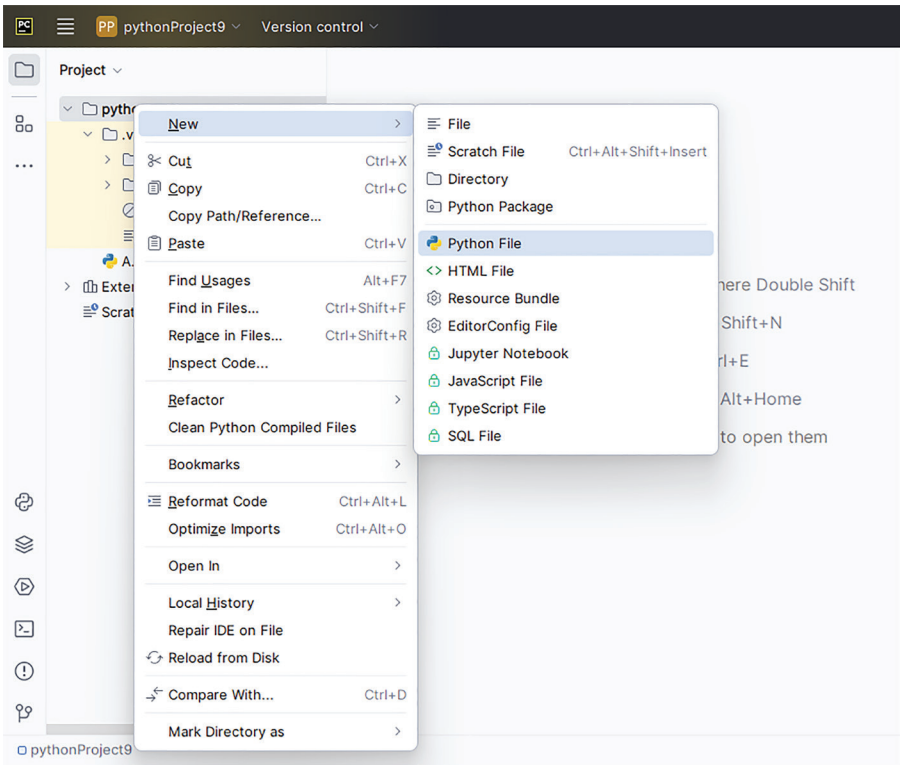
لإنشاء مشروع جديد، من قائمة File => new project



شكل (3-1) شاشة PyCharm Editor - New Project

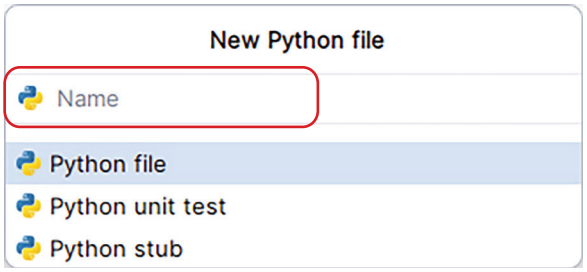
(9-1) إنشاء ملف بايثون جديد

يمكن أن يحتوي المشروع على العديد من ملفات التعليمات، ولإنشاء ملف بايثون جديد، من خلال الضغط بالزر الأيمن على اسم المشروع في منطقة (شجرة المشروع والملفات) ومن ثم اختيار python file <- New.



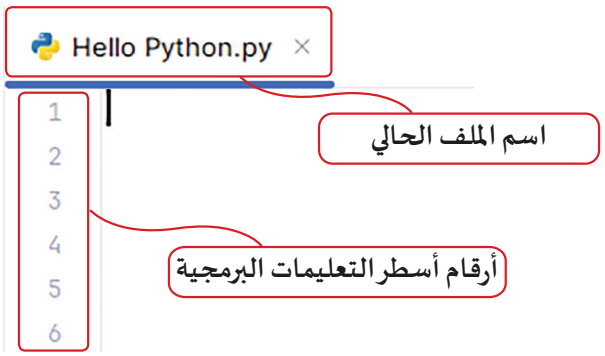
شكل (1-4) شاشة PyCharm Editor - New Python File

ثم كتابة اسم الملف



شكل (1-5) شاشة PyCharm Editor - File Name

تظهر منطقة تحرير التعليمات البرمجية، ويظهر في الأعلى علامة تبويب باسم الملف الحالي، ويظهر المؤشر في السطر البرمجي الأول.



شكل (1-6) شاشة PyCharm - Code Editor

(1-10) كتابة البرنامج الأول

في نافذة كتابة التعليمات البرمجية، وفي السطر الأول أكتب التعليمات البرمجية التالية، مع مراعاة الدقة.

```
1   print('I am a Python Developer')
```

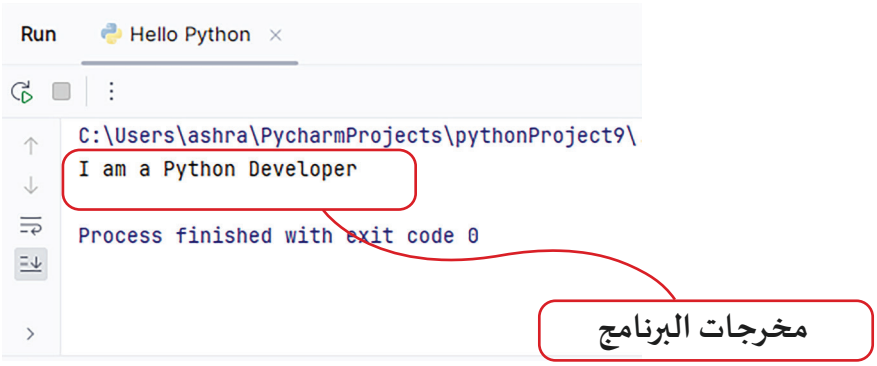
حيث التعليمة print هي من الدوال المدمجة في لغة بايثون والتي عند تنفيذ البرنامج تطبع المحتويات بين القوسين في نافذة التشغيل RUN.
لاحظ كتابة النص بين علامتي تنصيص، والتي سوف نوضح وظيفتها لاحقًا.

(1-11) تشغيل البرنامج

يمكن تشغيل البرنامج من خلال:

- القائمة المختصرة لمنطقة كتابة التعليمات البرمجية واختيار الأمر RUN.
- استخدام مفاتيح الاختصار Ctrl + Shift + F10 (تشغيل الملف الحالي).
- منطقة شريط الأدوات Toolbar واختيار الأداة RUN الموضحة في الشكل.

تظهر نافذة جديدة وبها نتيجة تنفيذ البرنامج كالتالي:



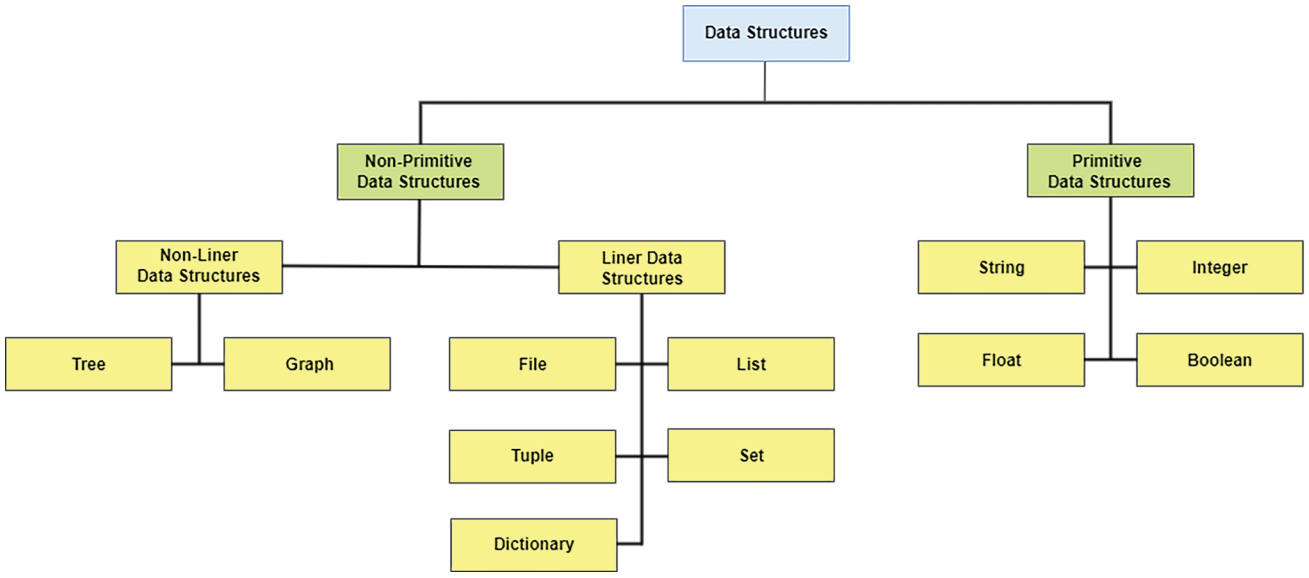
شكل (1-7) شاشة PyCharm Editor - RUN

2- هياكل البيانات Data Structure

طريقة لتنظيم وتخزين البيانات بشكل يسمح بالوصول إليها ومعالجتها بكفاءة. تُستخدم هياكل البيانات في جميع أنواع البرامج، من أبسطها إلى أكثرها تعقيدًا.

وتنقسم هياكل البيانات إلى قسمين أساسيين:

- أولًا: هياكل البيانات الأولية (Primitive)
- ثانيًا: هياكل البيانات غير الأولية (Non-Primitive)



الشكل (1-8) هياكل البيانات

(1-2) هياكل البيانات الأولية (Primitive):

- هي البنية التي تسمح بتخزين قيم لنوع واحد فقط من البيانات.
- يعتمد حجم هياكل البيانات الأولية على نوعها.
- تعد الأعداد الصحيحة والعشرية وسلاسل الحروف والبيانات المنطقية من هياكل البيانات الأولية.

أنواع هياكل البيانات الأولية:

- الأعداد الصحيحة Integer.
- الأعداد العشرية Float.
- السلسلة النصية String.
- البيانات المنطقية Boolean.

(2-2) هياكل البيانات غير الأولية (Non-Primitive):

- هي البنية التي تسمح بتخزين قيم أنواع بيانات متعددة.
- حجم بنية البيانات غير ثابت.
- تعد القائمة والمجموعة والقاموس والصف والمصفوفة من هياكل البيانات غير الأولية.

أنواع هياكل البيانات غير الأولية

- **البيانات الخطية:**

هي بنية يتم فيها تنظيم العناصر بشكل تسلسلي في الذاكرة، وقد يرتبط كل عنصر بالعنصر السابق والتالي، فيتم الوصول إليها في عملية تشغيل واحدة.

الشكل (1-9) Linear Data Structure

أ- القائمة List

تستخدم القوائم لتخزين مجموعة من القيم. يمكن أن تحتوي القوائم على أي نوع من البيانات، بما في ذلك الأعداد والسلسلة والقوائم الأخرى.

ب- المجموعة Set

تستخدم المجموعات لتخزين مجموعة من القيم الفريدة (Unique values). يمكن أن تحتوي المجموعات على أي نوع من البيانات، بما في ذلك الأعداد (Float - Intger) والسلسلة (string)، والقوائم الأخرى.

ج- القاموس Dictionary

تستخدم القواميس لتخزين مجموعة من المفاتيح والقيم. ويمكن أن تكون القيم من أي نوع من البيانات، بما في ذلك الأعداد والسلسلة.

د- الصف Tuple

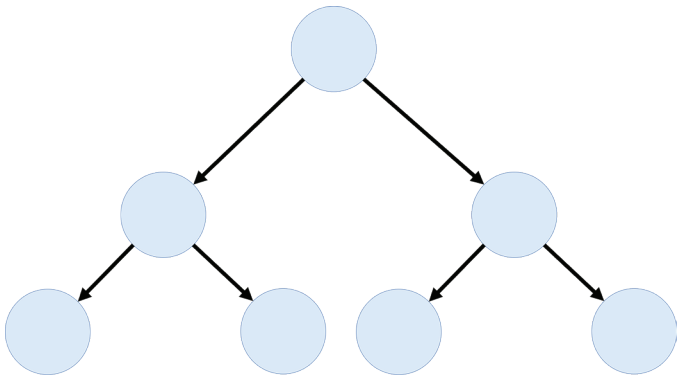
مجموعة من القيم غير قابلة للتعديل، حيث يمكن أن تحتوي على عناصر من أنواع مختلفة.

- **البيانات غير الخطية:**

بنية لا يتم فيها ترتيب عناصر البيانات بطريقة متجاورة، ولا يمكن الوصول إليها في عملية تشغيل واحدة.

a- Tree

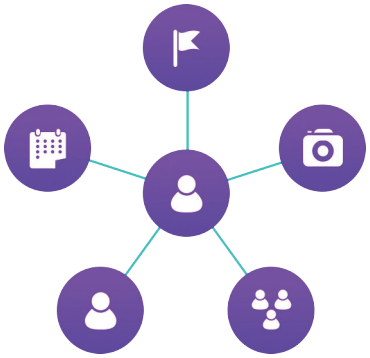
يحتوي هيكلها على شكل هرمي يمثل العلاقة بين عقد فرعية وعقد الأصل.



الشكل (1-10) Tree

هـ- Graph:

يحتوي هيكلها على عدد محدود من الرؤوس والحواف، تُستخدم الرؤوس لتخزين عناصر البيانات، بينما تمثل الحواف العلاقة بين الرؤوس.



الشكل (1-11) Graph

ورقة عمل (1-1)

برنامج حساب مساحة الدائرة

مشكلة البرنامج:

من خلال دراستك لقوانين مساحة الأشكال الهندسية، احسب مساحة الدائرة بمعلومية نصف القطر.

الخوارزمية:

- استقبل من المستخدم قيمة نصف قطر الدائرة.
- احسب مساحة الدائرة مستعينًا بالمعادلة التالية:

  مساحة الدائرة = πr^2، حيث ($\pi = 3.14$، و r نصف قطر الدائرة).
- اعرض مساحة الدائرة على الشاشة.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

ورقة عمل (1-2)

برنامج تحويل وحدات قياس درجات الحرارة

مشكلة البرنامج:

التحويل ما بين وحدات القياس المختلفة لدرجة الحرارة.

الخوارزمية:

- استقبل من المستخدم درجة حرارة الطقس.
- استقبل من المستخدم وحدة القياس الحالية.
- استخدم المعادلة التالية لتحويل درجة الحرارة من فهرنهايت إلى سيليزية: $°C = (°F - 32) \div 1.8$.
- استخدم المعادلة التالية لتحويل درجة الحرارة من سيليزية إلى فهرنهايت: $°F = (°C \times 1.8) + 32$.
- اعرض درجة الحرارة بعد التحويل لوحدة القياس المطلوبة.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

برمجة Python

المتغيرات Variables

نواتج التعلم

- التعرف على المتغيرات.
- تحديد أنواع المتغيرات.
- فهم قواعد تسمية المتغيرات.
- استخدام المتغيرات.
- استخدام دالة print().
- استخدام دالة input().
- التحويل بين أنواع البيانات.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

المتغيرات Variables

المتغيرات Variables

المتغير هو مكان يتم حجزه في الذاكرة لتخزين البيانات بصورة مؤقتة أثناء تنفيذ البرنامج، وله اسم، وقيمة، ونوع (الأعداد والنصوص والقوائم والقواميس، ...إلخ)، ولغة Python تُمكن من تغيير نوع وقيمة المتغير أثناء تنفيذ البرنامج.

(1-1) الإعلان عن المتغيرات Declaration

تتميز Python بالتعرف على نوع المتغير بشكل مباشر حسب القيمة المخصصة تلقائيًا، ويمكن تخصيص القيم للمتغيرات باستخدام رمز التخصيص (=).

(2-1) أنواع المتغيرات Data Types

نوع المتغير	نوع البيانات	استخداماته	مثال
string str( )	نصية	تخزين سلسلة من الأحرف والأرقام التي يتعامل معها كنصوص، ولابد أن تكتب بين علامتي التنصيص المزدوجة " " أو المفردة ' '.	main.py 1 my_name = "Ali" 2 my_country = 'Kuwait' 3 my_address = '15 Beirut street'
Integer int( )	عددية صحيحة	تخزين قيم عددية صحيحة.	main.py 1 months_count = 12
float float( )	عددية عشرية	تخزين قيم تحتوي على كسور عشرية.	main.py 1 my_salary = 1930.00
boolean bool( )	منطقية	وتحتمل إحدى قيمتين فقط (True أو False)، وغالبًا ما تنتج عن عمليات المقارنة.	main.py 1 var_num1 = 10 > 5 # True 2 var_num2 = 10 < 5 # False

جدول (1-1) أمثلة على أنواع البيانات

كما وجد أنواع أخرى من البيانات ستتعرف عليها في وقت لاحق مثل القوائم والصفوف.

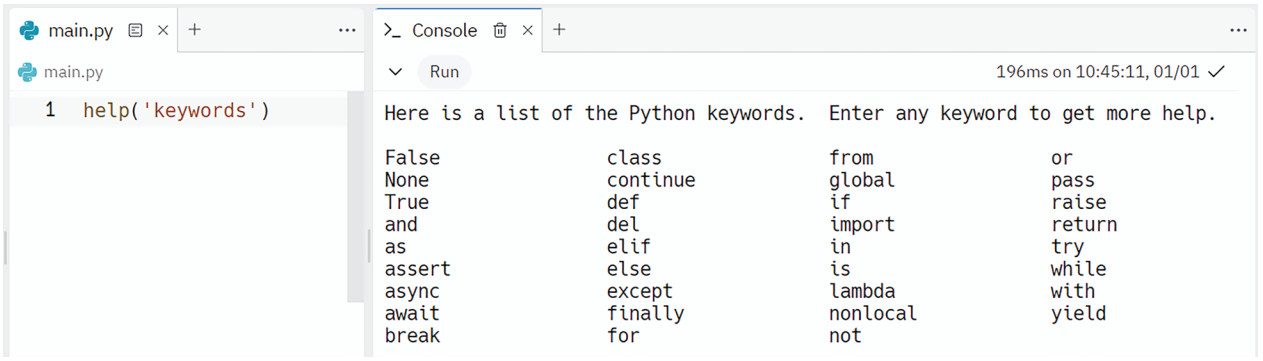
(3-1) قواعد تسمية المتغيرات Naming Variables.

يجب أن تتبع المتغيرات في Python قواعد معينة للتسمية:

- يجب أن يبدأ اسم المتغير (بحرف أو شرطة سفلية)، ولا يبدأ برقم.
- يمكن أن يحتوي اسم المتغير على أحرف أبجدية وأرقام.
- لا يمكن أن يحتوي اسم المتغير على مسافات أو رموز خاصة (!@#$%^&*) باستثناء الشرطة السفلية (_).
- لا يمكن استخدام الكلمات المحجوزة reserved keywords لتسمية المتغيرات.

ملاحظة: يراعى في تسمية اسم المتغير حالة الأحرف case letters for sensitive (Capital / Small) .

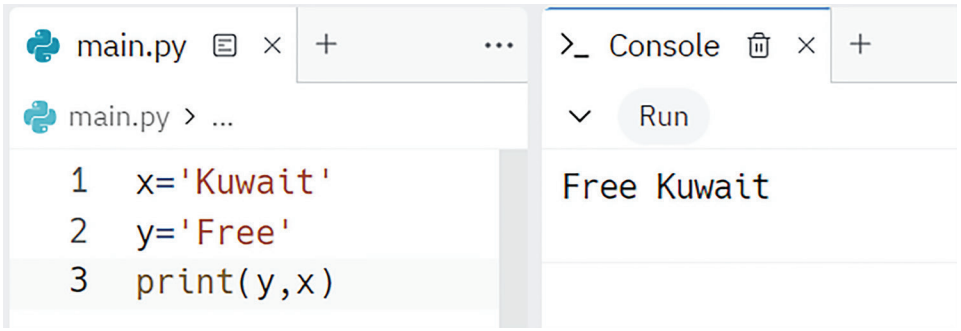
للتعرف على الكلمات المحجوزة نستخدم الأمر help('keywords')



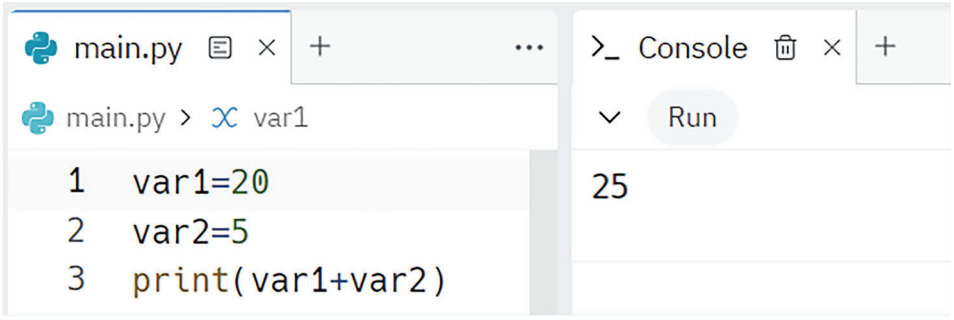
(4-1) دالة الإخراج Print

دالة print هي دالة مدمجة في Python، وتستخدم لطباعة البيانات على الشاشة. يمكن استخدامها لطباعة النصوص، والأرقام، والقوائم، وغيرها. وفي الجدول التالي بعض أشكال استخدامات دالة ()print.

مثال (1-1)

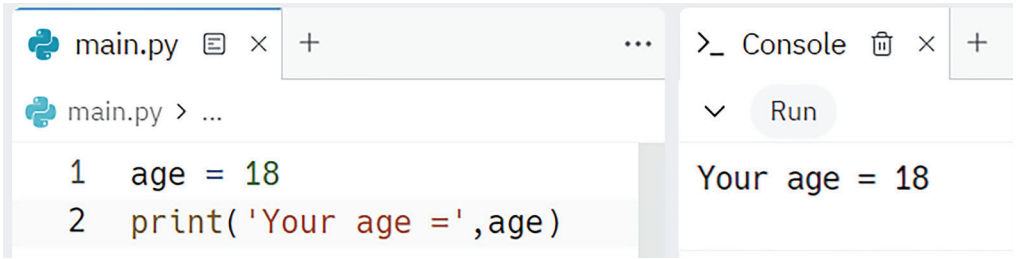


مثال (1-2)



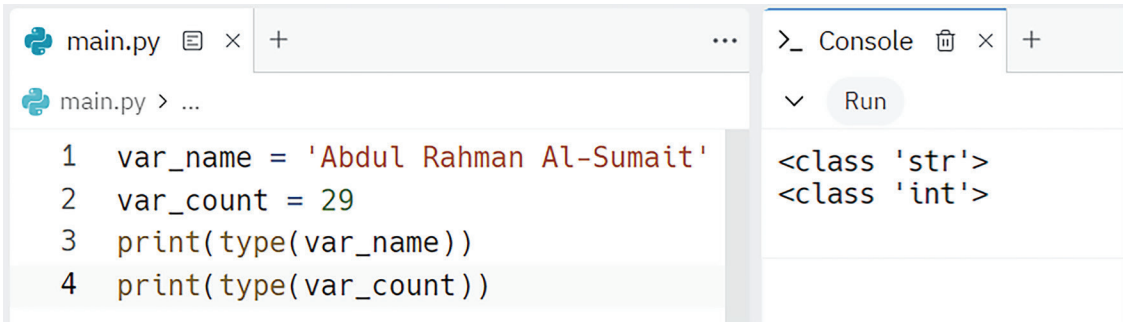
- تم استخدام عملية الجمع كأحد أنواع العمليات الحسابية، والتي سيتم التطرق لها لاحقًا.

مثال (1-3)



لمعلوماتك

التعرف على نوع المتغير الدالة ()epyt:



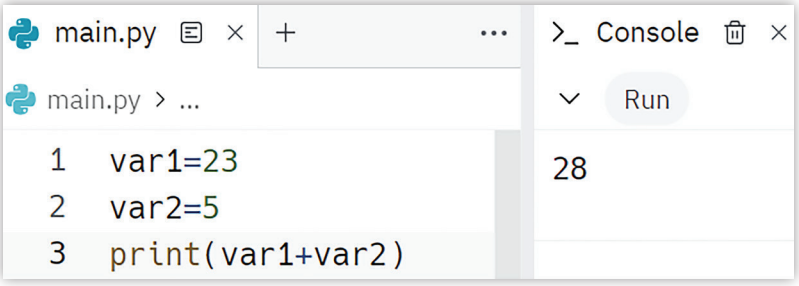
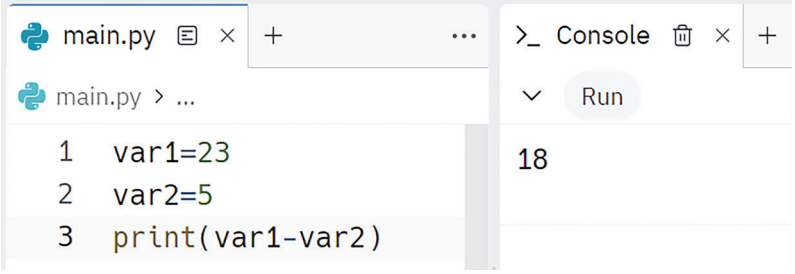
لاحظ:

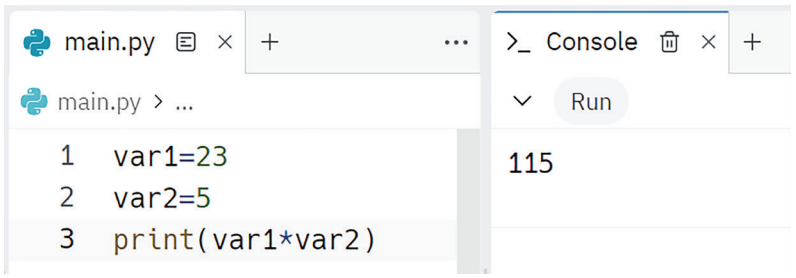
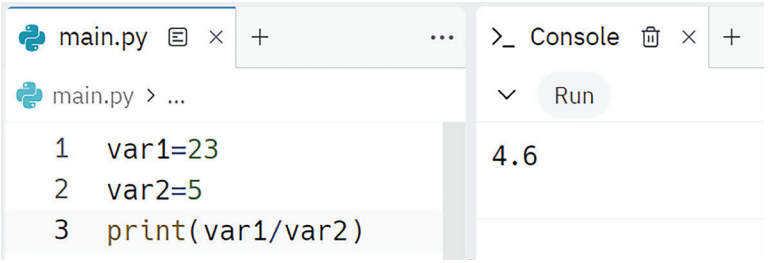
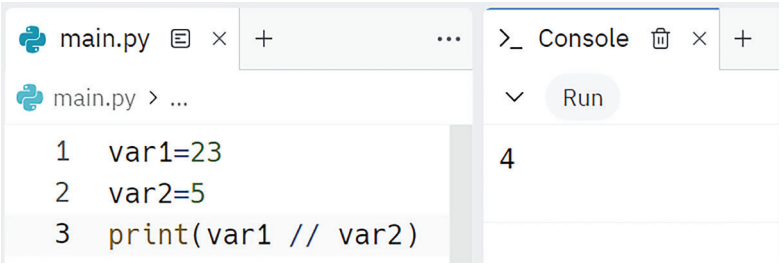
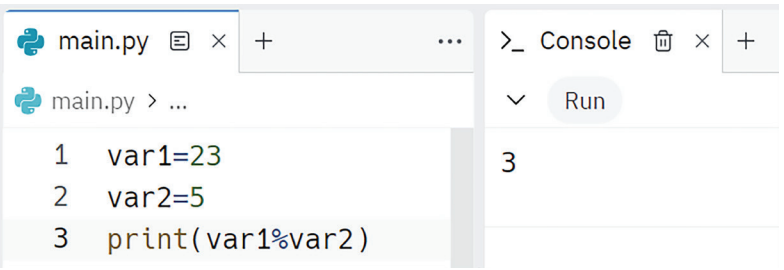
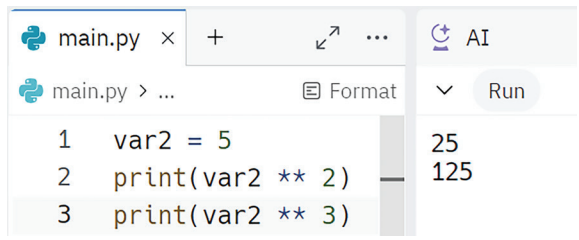
ظهور نوع المتغير الأول eman_rav من نوع نصي <'rts' ssalc>

ظهور نوع المتغير الثاني tnuoc_rav من نوع عدد صحيح <'tni' ssalc>

(5-1) العمليات الحسابية

تستخدم Python العمليات الحسابية الأساسية، وفي الجدول التالي رموز العمليات الحسابية ووظيفتها.

الرمز	العمليات الحسابية	مثال
+	تستخدم لعملية الجمع Addition	
-	تستخدم لعملية الطرح Subtraction	

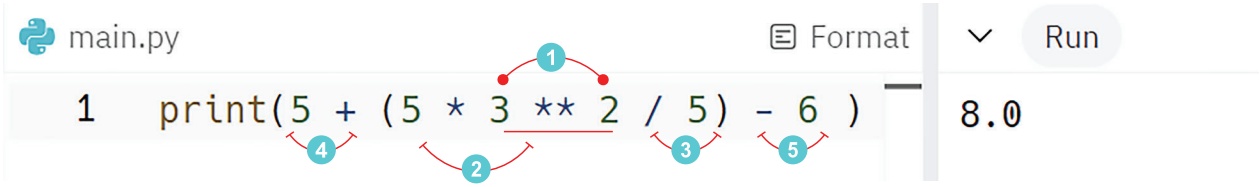
الرمز	العمليات الحسابية	مثال
*	تستخدم لعملية الضرب Multiplication	
/	تستخدم لعملية القسمة Division	
//	تستخدم لإيجاد ناتج القسمة الصحيح بدون الكسور Floor division	
%	تستخدم لإيجاد باقي القسمة Modulus	
**	لرفع العدد إلى أس محدد Exponentiation	

جدول (1-2) رموز العمليات الحسابية ووظيفتها

(6-1) أولويات تنفيذ العمليات الحسابية

1- العمليات داخل الأقواس.
2- رفع الأسس.
3- الضرب والقسمة.
4- الجمع والطرح.
5- يتم تنفيذ العمليات المتكافئة من اليسار إلى اليمين.

مثال (1-4)



(7-1) دالة input()

هي دالة مدمجة في Python تُستخدم لجمع البيانات المُدخلة من المستخدم، مثل أسماء المستخدمين وعناوين البريد الإلكتروني وأرقام الهواتف.

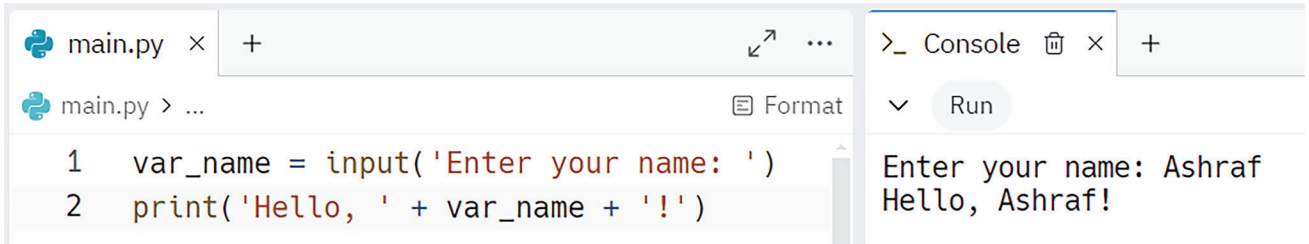
صيغة الدالة Syntax

تكتب صيغة الدالة input() كالتالي:

input([prompt])

prompt (اختياري): سلسلة نصية تُستخدم لعرض رسالة إلى المستخدم.

مثال (1-5)



ملاحظة:

يستخدم المعامل + لتجميع السلاسل النصية strings.

تفسير النص البرمجي:

1- يعرض رسالة للمستخدم Enter your name:

2- ينتظر البرنامج حتى يُدخل المستخدم اسمًا ويضغط على مفتاح Enter.

3- تخصيص القيمة المدخلة إلى المتغير var_name.

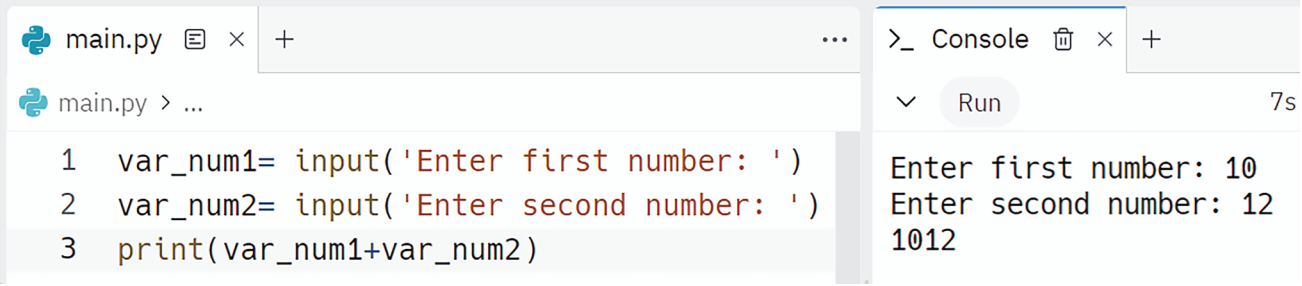
4- طباعة العبارة Hello!, [Name].

لاحظ:

- يجب الانتباه إلى أن دالة input() تُرجع سلسلة string. ويمكن تحويلها إلى عدد صحيح أو عدد عشري، باستخدام الدالة int() أو float().
- كما يمكن التحويل بين أنواع البيانات باستخدام الدوال bool() – str() .

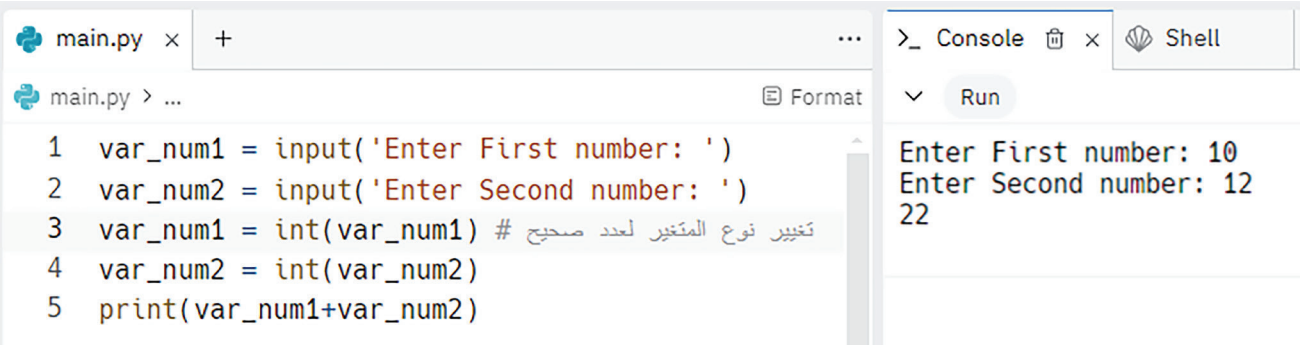
مثال (1-6)

يتم تجميع المتغيرين الأول والثاني بجانب بعضهما كنصوص ولم يجمعهما كأرقام.



مثال (1-7)

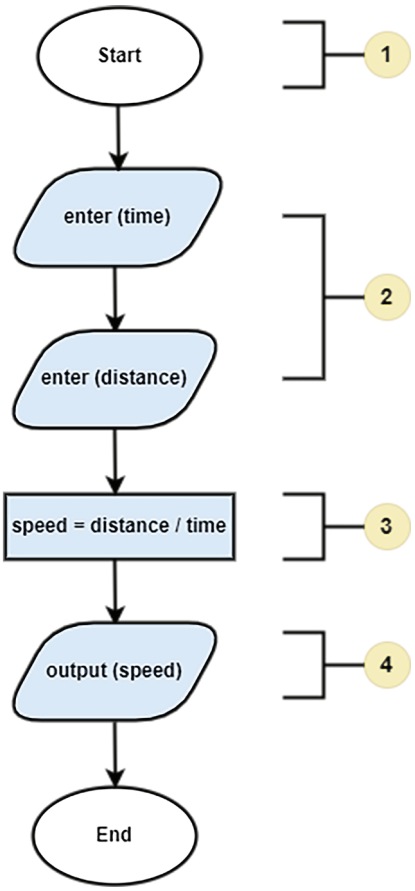
يتم إجراء عملية حسابية بجمع المتغيرين الأول والثاني كأرقام.



كما يمكن التحويل مباشرة أثناء عملية الإدخال كالتالي:

ناتج استخدام الدالة input()، والدالة int() تُعيد العدد المدخل إلى عدد صحيح.

مثال تطبيقي (1-8)



شكل (1-1) خريطة تدفق مثال تطبيقي

من خلال مصادر التعلم المتاحة، ادرس خريطة التدفق، التي تمثل قانون حساب (السرعة العددية) الذي درسته في منهج الفيزياء، ثم ناقش مع معلمك وزملائك كيفية تحويل خريطة التدفق إلى برنامج مكتوب بلغة Python.

1- يمثل بداية البرنامج.

2- يطلب من المستخدم إدخال قيمتي المسافة والزمن، ولتنفيذ ذلك باستخدام دالة ()input.

أ. التعليمة البرمجية لإدخال المسافة distance

- ظهور رسالة ('Enter the distance (km)') للمستخدم عند تنفيذ البرنامج.
- استقبال قيمة من المستخدم باستخدام دالة ()input.

```
var_distance = input('Enter the distance (km)')
```

ب. يمكنك كتابة التعليمة البرمجية اللازمة لإدخال قيمة الزمن time.

3- حساب السرعة تستخدم معادلة السرعة القياسية التالي: speed = distance / time

- يتم الإعلان عن متغير جديد باسم var_speed، ثم تخصيص قيمه له وهي (ناتج قسمة المسافة على الزمن).
- المعامل (/) يمثل عملية القسمة، راجع جدول العمليات الحسابية.
- اخترإحدى التعليمات البرمجية التالية لحساب السرعة:

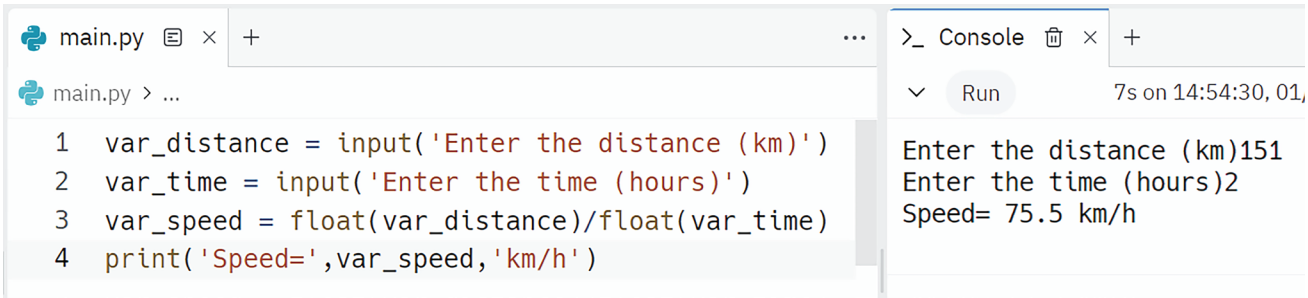
يمكن استخدام هذه التعليمة البرمجية بإضافة دالةfloat() مع دالة input.

```
var_speed = float(var_distance)/float(var_time)
var_speed = float(var_distance/var_time)
var_speed = var_distance/var_time
```

- تم استخدام دالةfloat() لتحويل قيمة المسافة والزمن لقيمة عددية عشرية.

4- تمثل عملية الإخراج.

5- نهاية البرنامج.



ورقة عمل (1-1)

برنامج حساب العجلة Acceleration

مشكلة البرنامج:

حساب العجلة بمعلومية محصلة القوى والكتلة، وفقًا لقانون نيوتن الثاني.

الخوارزمية:

- استقبل من المستخدم القوى المؤثرة Force بالنيوتن.
- استقبل من المستخدم الكتلة Mass بالكيلو جرام، (أكبر من الصفر).
- احسب العجلة Acceleration باستخدام القانون الثاني لنيوتن

العجلة = محصلة القوى / الكتلة

$$\text{Acceleration } (m/s^2) = \text{ Force } (N) / \text{ mass } (kg)$$

- اطبع قيمة العجلة على الشاشة.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- أنشيء ملف Python باسم Acceleration.py.
- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

ورقة عمل (1-2)

برنامج حساب قوة الشغل المبذول

مشكلة البرنامج:

حساب الشغل المبذول بمعلومية القوى المؤثرة، والإزاحة في اتجاهها.

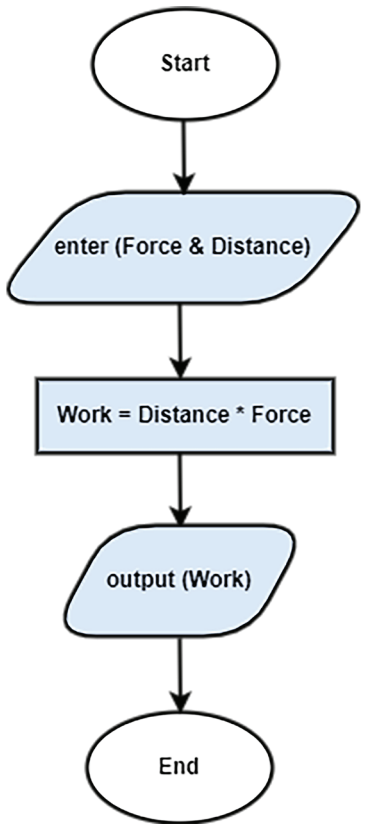
الخوارزمية:

- استقبل من المستخدم القوى المؤثرة Force بوحدة النيوتن.
- استقبل من المستخدم الإزاحة Distance بوحدة المتر، (أكبر من الصفر).
- احسب البرنامج الشغل Work بوحدة الجول، باستخدام المعادلة الشغل W = القوة F × الإزاحة D
- اطبع ناتج المعادلة (الشغل) على الشاشة.

المطلوب:

خريطة التدفق

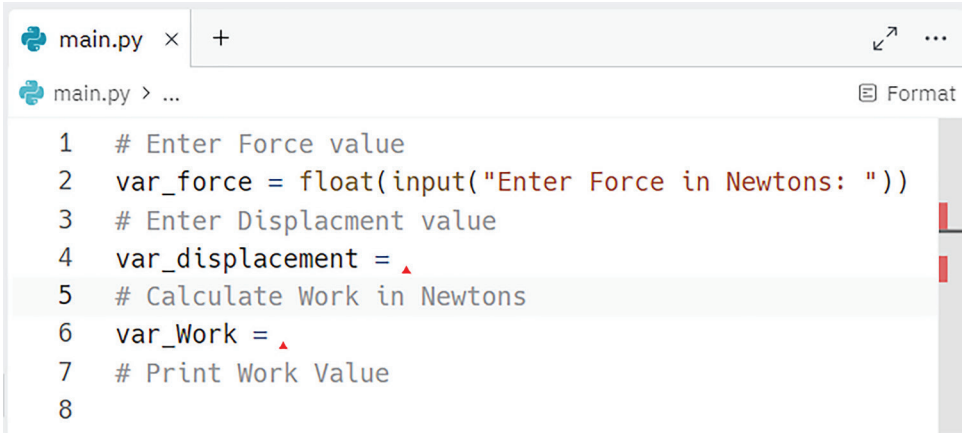
- افتح الملف work.py.
- ادرس خريطة التدفق المقابلة.
- أضف التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.



شكل (1-2) خريطة تدفق الشغل

البرنامج

- افتح الملف work.py.
- أكمل التعليمات البرمجية اللازمة ليعمل البرنامج بشكل صحيح.
  - المتغير var_force يمثل القوى المؤثرة.
  - المتغير var_distance يمثل الإزاحة.
  - المتغير var_work يمثل الشغل.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.



برمجة Python

الشروط Conditions

نواتج التعلم

- يميز بين عمليات الإسناد والعمليات المنطقية.
- يوظف العمليات الشرطية if, else, elif لاتخاذ القرارات.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

الشروط Conditions

بنية تحكم تستخدم لاتخاذ القرارات وتنفيذ التعليمات بناءً على تحقق شرط محدد، والتحكم في مسار البرنامج.

(1-1) عمليات المقارنة

العمليات التي تقارن بين عاملين (متغيرات، قيم، ...)، والنتيجة تكون من نوع boolean إما True أو False.

المعامل	الوصف
==	يساوي
!=	لا يساوي
<	أصغر من
>	أكبر من
<=	أصغر من أو يساوي
>=	أكبر من أو يساوي

جدول (1-1) معاملات المقارنة

لاحظ:

معامل = يعني تخصيص قيمة للمتغير Assignment.

معامل = = يعني المقارنة بين عنصرين Comparison.

مثال (1-1)

main.py

```
1  # Assignment
2  var_number = 50
3  # Comparison
4  print(var_number==70) #false
5  print(var_number==50) #true
```

لمعلوماتك

x	y	x == y	x != y	x > y	x < y	x >= y	x <= y
1	1	TRUE	FALSE	FALSE	FALSE	TRUE	TRUE
1	2	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE
2	1	FALSE	TRUE	TRUE	FALSE	TRUE	FALSE

جدول (1-2) جدول نتائج عمليات المقارنة

- تُستخدم التعليمة البرمجية var_x % 2 == 0 لتحديد أن العدد الصحيح الذي يمثل قيمة المتغير var_x زوجيًا.
- تُستخدم التعليمة البرمجية var_x % 2 == 1 لتحديد أن العدد الصحيح الذي يمثل قيمة المتغير var_x فرديًا.

(2-1) العمليات المنطقية

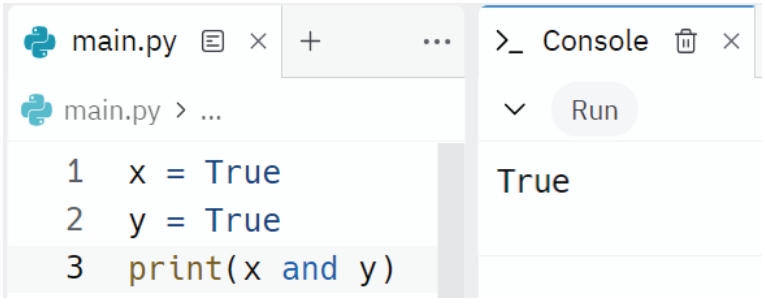
تُستخدم لإجراء عمليات منطقية على قيم المتغيرات، كما بالجدول (1-3):

x	y	x and y	x or y	not x
True	True	True	True	False
True	False	False	True	
False	True	False	True	True
False	False	False	False	

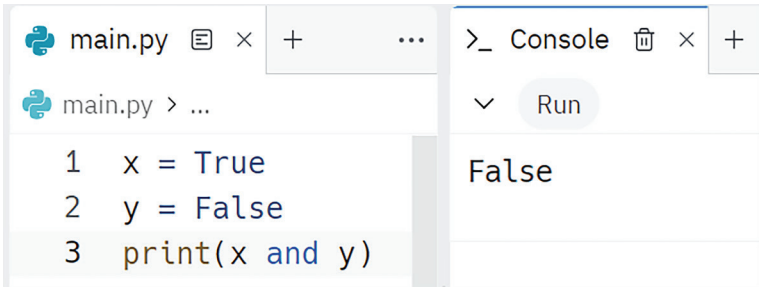
جدول (1-3) العمليات المنطقية

- العملية المنطقية and: تُرجع True إذا كانت كلتا القيمتين المنطقيتين صحيحتان، وإلا تُرجع False.

مثال (1-2)

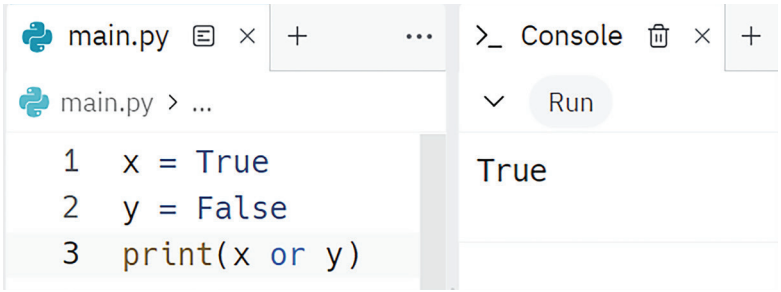


مثال (1-3)



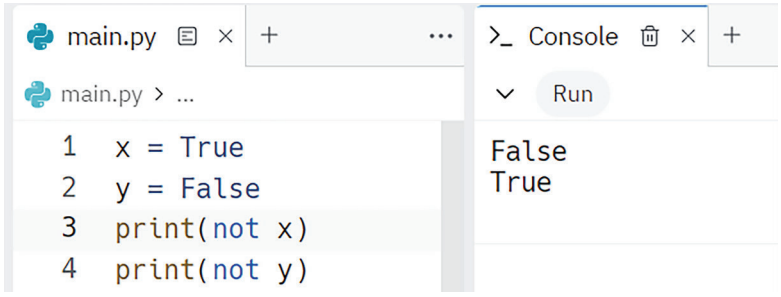
- العملية المنطقية or: تُرجع True إذا كانت أي من القيمتين المنطقيتين صحيحتان، وإلا تُرجع False.

مثال (1-4)



- العملية المنطقية NOT: تُرجع False إذا كانت القيمة المنطقية صحيحة، وإلا تُرجع True.

مثال (1-5)



مثال (1-6)

لكتابة شرط أن درجة الطالب تتراوح من 0 إلى 100.

x >= 0 and x<=100

(3-1) العمليات الشرطية

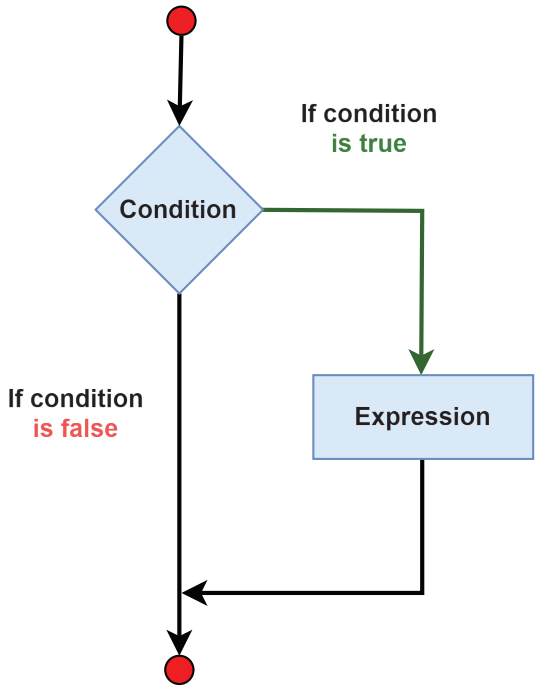
يمكن استخدام العمليات الشرطية if و elif و else لاتخاذ القرارات وتنفيذ التعليمات بناءً على نتائج تلك القرارات.

(1-3-1) العملية الشرطية if

تُستخدم الدالة الشرطية if لاختبار شرط منطقي، وتنفيذ تعليمات برمجية إذا تحقق الشرط.

الصيغة العامة Syntax:

```
if condition:
    expression
    expression
    expression
```



شكل (1-1) خريطة تدفق الصيغة العامة fi

التعليمات البرمجية التي تأخذ أكثر من سطر تأخذ نفس المسافة البادئة (indentation).

```
main.py
main.py > ...
Format
1  prog_lan = input('What is your favorite programming language? ')
2  if prog_lan == 'python':
3    print('Python is my favorite too!')
4
```

لاحظ:

يستخدم Python المسافة البادئة بينما تستخدم بعض اللغات الأخرى الأقواس والعبارات للدلالة على البداية والنهاية.

مثال (1-7)

برنامج يُظهر تقدير الطالب (امتياز) في حال حصوله على درجة الامتياز في أحد الاختبارات.

```
main.py
main.py > ...
1  var_grade = int(input('Enter student degree'))
2  if var_grade >= 90:
3  print('You got an excellent grade')
4  print('Good luck')
```

indentation

في المثال السابق ستظهر رسالة خطأ عند تنفيذ البرنامج حيث لا يوجد تعليمة برمجية يتم تنفيذها عند تحقق الشرط، والصحيح أن تكتب التعليمات البرمجية كالتالي:

```
main.py
1  var_grade = int(input('Enter student degree '))
2  if var_grade >= 90:
3    print('You got an excellent grade')
4  print('Good luck')
```

```
Console
Run   5s on 22:53:32, 01/03
Enter student degree 95
You got an excellent grade
Good luck

Run   6s on 22:55:53, 01/03
Enter student degree 80
Good luck
```

إذا تحقق الشرط سيطبع عبارة «You got an excellent grade» ثم عبارة «Good luck».

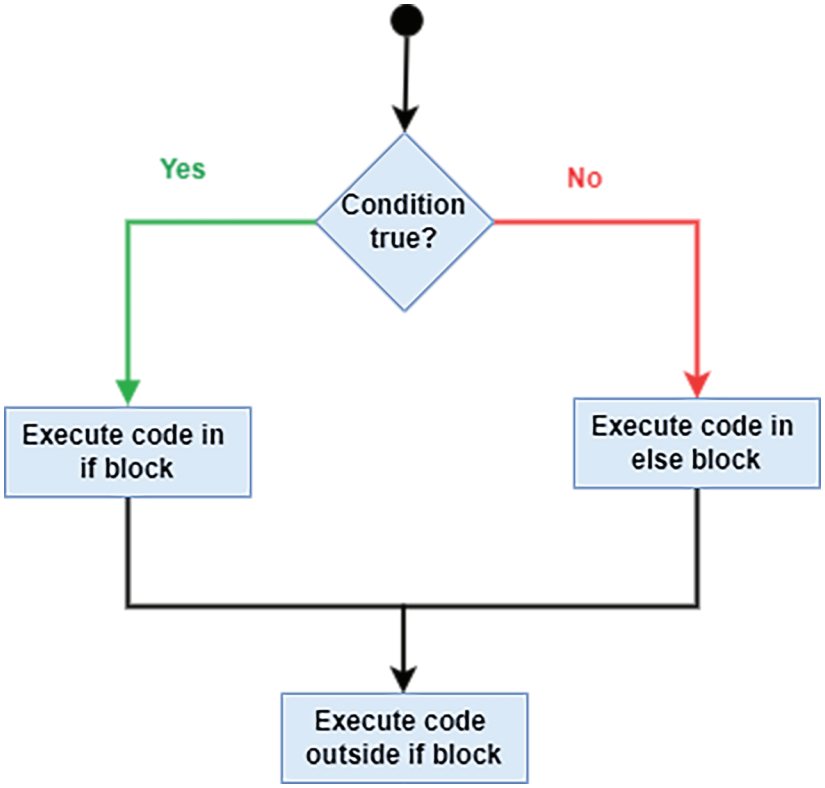
إذا لم يتحقق الشرط سيتجاهل تعليمة fi، وينتقل للتعليمة البرمجية التالية print ليطبع عبارة «Good luck».

(2-3-1) العملية الشرطية else

تستخدم لتحديد ما يجب تنفيذه إذا لم يتحقق الشرط.

الصيغة العامة Syntax:

```
if condition:
   expression
   expression
else:
   expression
   expression
```



شكل (2-1) خريطة تدفق الصيغة العامة else

التعليمات البرمجية التي تأخذ أكثر من سطر تأخذ نفس المسافة البادئة (indentation).

```
main.py
1  a = input('Enter the easiest programming language: ')
2  if a == 'Pyhton':
3    print('Yes! You are right')
4  else:
5    print('Nope! You are wrong')
```

مثال (1-8)

برنامج يُظهر حصول الطالب على تقدير امتياز في أحد الاختبارات بناءً على الدرجة المدخلة.

```
main.py
1  var_grade = int(input('Enter exam degree '))
2  if var_grade >= 90:
3    print('You got an excellent grade')
4  else:
5    print('Work hard to get an excellent grade')
```

```
Console
Run                                   9s on 23:15:01, 01/03
Enter exam degree 95
You got an excellent grade

Run                                   2s on 23:15:12, 01/03
Enter exam degree 60
Work hard to get an excellent grade
```

مثال (1-9) إثرائي

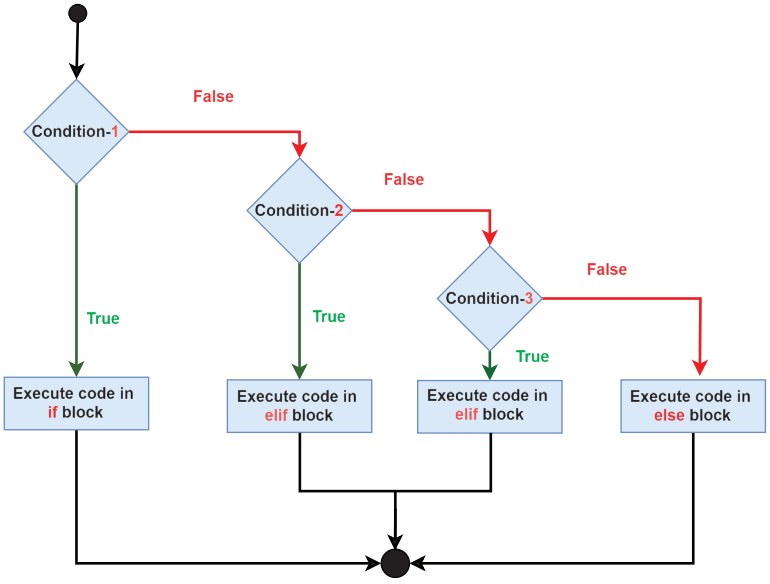
طور برنامج لإدخال درجة طالب بحيث تظهر رسالة خطأ عند إدخال درجة غير صحيحة، علمًا بأن درجة الطالب تتراوح بين 0 و100.

(3-3-1) العملية الشرطية elif

وهي اختصارلجملة (else if) وتستخدم لإضافة شروط متعددة إلى العملية الشرطية if. ويتم تنفيذ النص البرمجي الموجود داخل كتلة التعليمات البرمجية إذا تحقق شرط من الشروط.

الصيغة العامة Syntax:

```
if condition1:
   expression if condition1 is true.
elif condition2:
   expression if condition2 is true.
elif condition3:
   expression if condition3 is true.
...
else:
   expression if all conditions are false.
```



شكل (3-1) خريطة تدفق الصيغة العامة elfi

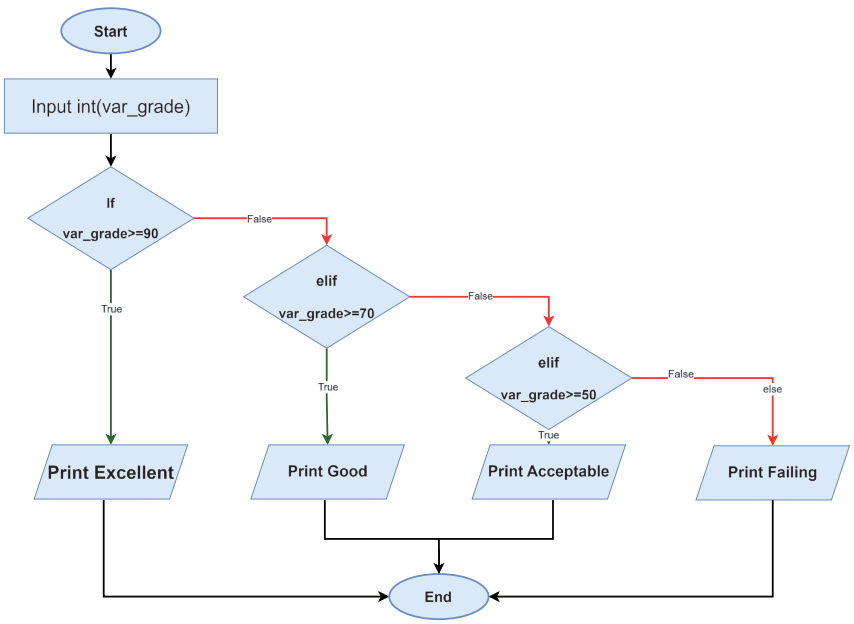
مثال (1-10)

برنامج يُظهر تقدير الطالب في أحد الاختبارات بناءً على الدرجة المدخلة.

main.py

```
1  var_grade = int(input("Enter exam grade: "))
2  if var_grade >= 90:
3      print('You got an Excellent')
4  elif var_grade >= 70:
5      print('You got Good')
6  elif var_grade >= 50:
7      print('You got an Acceptable')
8  else:
9      print('You have failed')
```

وفي المثال السابق إذا تحقق الشرط الأول سينفذ البرنامج التعليمات الخاصة به ويتخطى اختباراقي الشروط، وإذا لم يتحقق سيختبر الشرط الثاني، فإذا تحقق سينفذ التعليمات الخاصة به، وهكذا .



شكل (1-4) خريطة تدفق برنامج تقدير الطالب

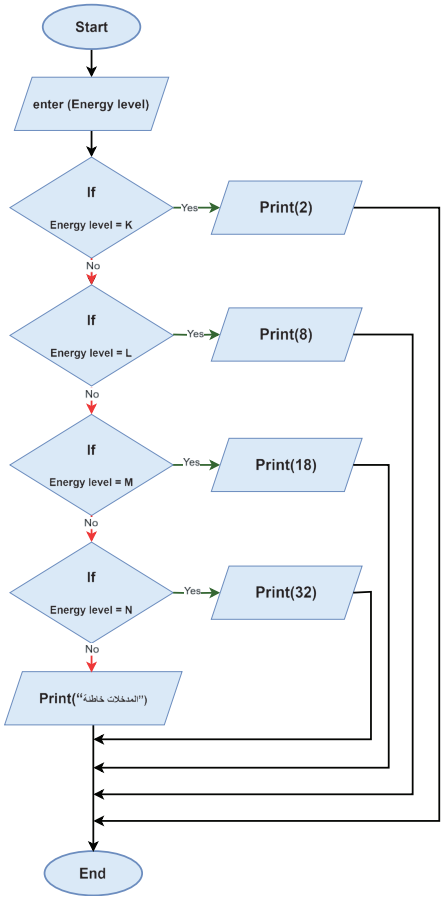
مثال تطبيقي (1-11)

من خلال دراستك لمادة الكيمياء، لمعرفة العدد الأقصى من الإلكترونات التي توجد في كل مستوى طاقة في الذرة من الجدول التالي:

رقم مستوى الطاقة	الأول	الثاني	الثالث	الرابع
الرمز	K	L	M	N
أقصى عدد من الإلكترونات	2	8	18	32

جدول (1-4) العدد الأقصى من الإلكترونات التي توجد في كل مستوى طاقة في الذرة

ادرس خريطة التدفق التالية ثم افتح الملف (energylevel.py)، واستكمل البرنامج المقترح والذي يُظهر العدد الأقصى للإلكترونات حسب رمز مستوى الطاقة، ليعمل بشكل صحيح.



شكل (1-5) خريطة تدفق برنامج التعرف على العدد الأقصى للإلكترونات


main.py
main.py > ...
1 energylevel = input('أدخل رمز مستوى الطاقة: ')
2 if(energylevel == 'K'):
3 print(2)
4
5
6
7
8
9 else:
10 print('المدخلات خطأ')

ورقة عمل (1-1)

برنامج تحويل وحدات قياس المسافة المختلفة

مشكلة البرنامج:

تحويل وحدة قياس المسافة من الكيلو متر km إلى المتر m والعكس.

الخوارزمية:

- استقبل من المستخدم المسافة Distance.
- استقبل من المستخدم وحدة القياس Unit.
- استخدم المعادلة المناسبة وفقًا لوحدة القياس المُدخلة:

- المسافة بالكيلو متر = المسافة بالمتر * 1000، (للتحويل من متر إلى كيلو متر).

- المسافة بالمتر = المسافة بالكيلو متر / 1000، (للتحويل من كيلو متر إلى متر).

- اطبع المسافة بوحدة القياس المطلوبة.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- أنشيء ملف Python باسم Unit_converter.py.
- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

الشروط Conditions

خريطة التدفق

ورقة عمل (1-2)

برنامج التعرف على حالة الأعداد

مشكلة البرنامج:

التعرف على حالة الأعداد المُدخلة من المستخدم.

الخوارزمية:

- استقبل من المستخدم العدد number.
- استخدم الشروط Conditions:

- إذا كان العدد المُدخل أكبر من صفر، يطبع (العدد موجبًا).
- إذا كان العدد المُدخل أصغر من صفر، يطبع (العدد سالبًا).
- إذا كان العدد المُدخل يساوي صفر، يطبع (صفر).

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

الشروط Conditions

خريطة التدفق

ورقة عمل (1-3)

برنامج تصميم آلة حاسبة

مشكلة البرنامج:

استخدام المعاملات الحسابية (+ ، - ، * ، /) في تنفيذ العمليات الحسابية (الجمع، الطرح، الضرب، القسمة).

الخوارزمية:

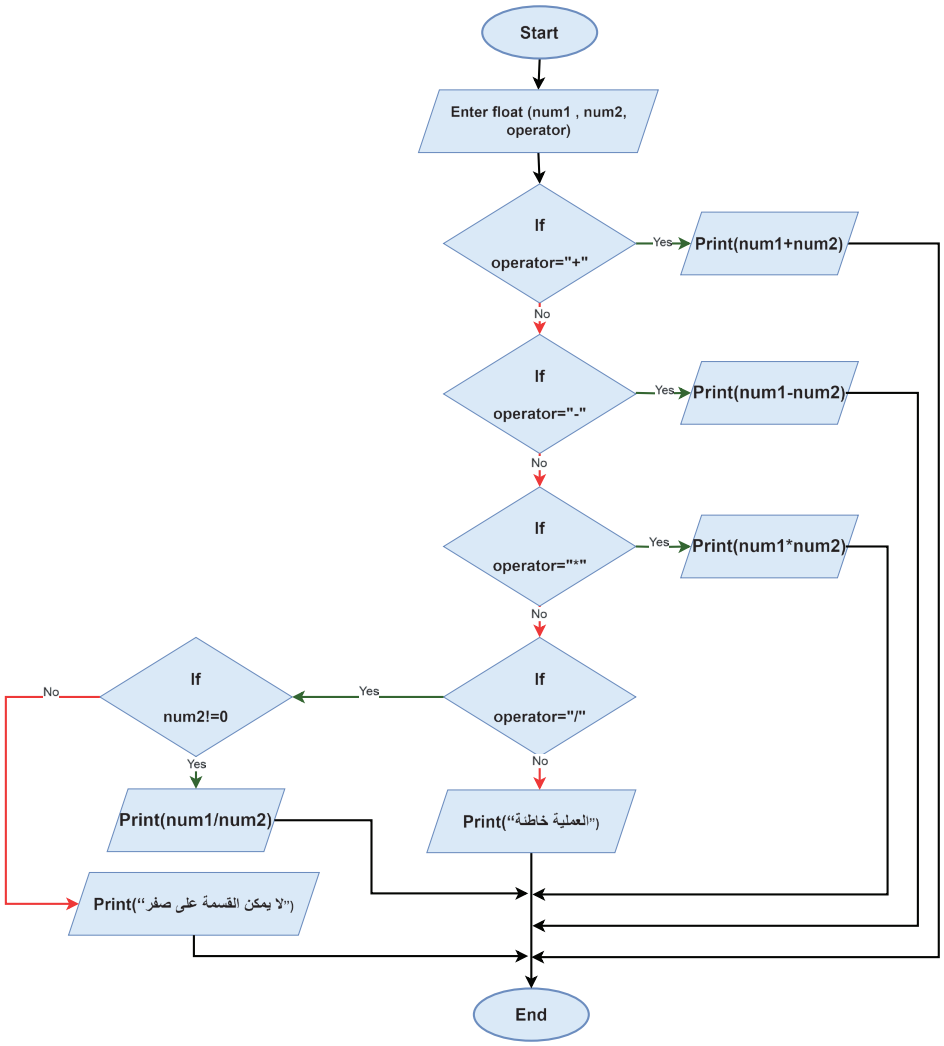
- استقبل من المستخدم العدد الأول، ثم العدد الثاني.
- استقبل من المستخدم المعامل الحسابي المطلوب.
- استخدم الشروط Conditions:
  - إذا كان المعامل المُدخل +، يجمع العددين الأول والثاني.
  - إذا كان المعامل المُدخل -، يطرح العدد الثاني من العدد الأول.
  - إذا كان المعامل المُدخل *، يضرب العددين الأول والثاني.
  - إذا كان المعامل المُدخل /، يقسم العدد الأول على العدد الثاني.

  (التأكد من العدد الثاني لا يساوي صفرًا، وطباعة رسالة تفيد ذلك).
- طباعة ناتج العملية الحسابية.

المطلوب:

خريطة التدفق

- ادرس خريطة التدفق التالية، والتي صممت لعمل آلة حاسبة بسيطة للعمليات الأساسية.



شكل (1-6) خريطة تدفق برنامج آلة حاسبة بسيطة للعمليات الأساسية

البرنامج

- افتح الملف Calculator.py، ثم نفذ التعديلات المناسبة ليعمل البرنامج بشكل صحيح.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

main.py

```
# Calculator Program
# Enter first, second number and math. operator.
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
operator = input("Enter math. operator (+, -, *, /): ")
# Conditions and math. operator
if operator == '+':
    print(num1 + num2)


elif operator == '*':
    print(num1 * num2)
elif operator == '/':



else:
    print("Invalid operator")
```

برمجة Python

التكرار Loops

نواتج التعلم

- التعرف على مفهوم التكرار.
- التعامل مع الدالة التكرارية while.
- التعامل مع الدالة التكرارية for.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

التكرار

هو تنفيذ مجموعة من العبارات بشكل متكرر، وسنتناول دوال التكرار التالية:

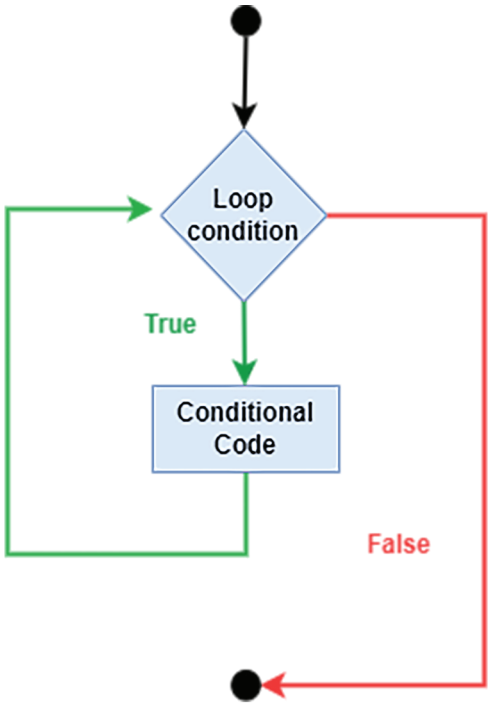
- الحلقة التكرارية: **while**.
- الحلقة التكرارية: **for**.

(1-1) الحلقة التكرارية while

تستخدم لتكرار النص البرمجي طالما تحقق شرط محدد (تكرار غير محدد).

الصيغة العامة Syntax:

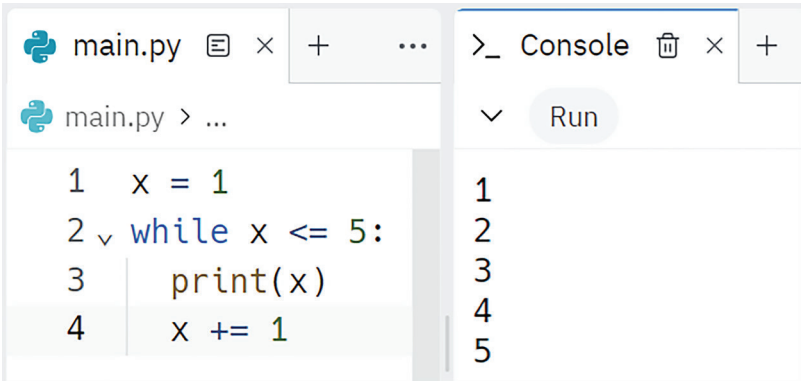
```
while condition:
    code block to be executed until the condition becomes False.
```

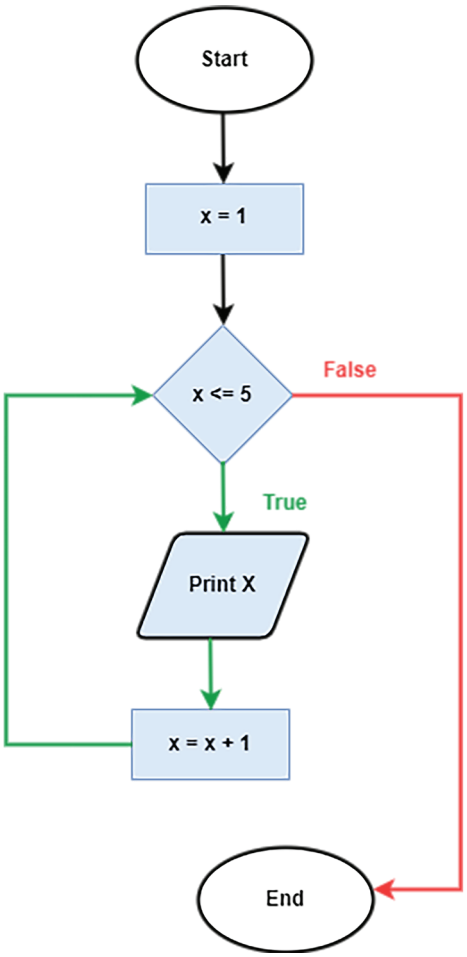


شكل (1-1) خريطة تدفق الصيغة العامة while

مثال (1-1)

يمكننا استخدام الدالة while لطباعة الأعداد من 1 إلى 5 على النحو التالي:





شكل (1-2) خريطة تدفق طباعة الأعداد

عند استخدام العبارة while هناك ثلاث نقاط يجب الانتباه لها:

- الشرط: غالبًا ما يتضمن الشرط متغيرًا.
- القيمة الابتدائية: يجب إعطاء قيمة ابتدائية للمتغير قبل بداية تنفيذ العبارة while بحيث تكون قيمة الشرط صحيحة.
- الخروج من التكرار: يجب أن تتضمن العبارات داخل تعليمات التكرار عبارة تسمح بإنهاء التكرار، وتكون عبارة تخصيص للمتغير بحيث تكون قيمة الشرط خطأ أو عبارة break.

لاحظ

التكرارات اللانهائية Infinity loops

تستخدم while True لتكرار تنفيذ التعليمات البرمجية إلى ما لا نهاية، على النحو التالي:

main.py

```
1 while True:
2     print('Hello, world!')
```

Run

```
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
Hello, world!
```

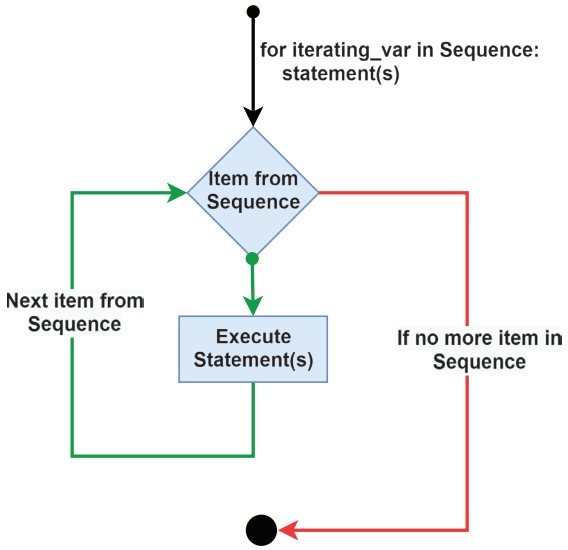
- ستؤدي هذه التعليمات البرمجية إلى طباعة Hello, world دون توقف.
- يمكن إيقاف تنفيذ التعليمة البرمجية باستخدام أداة Stop.
- عدل التعليمات البرمجية في المثال **(1-1)** لطباعة قيمة المتغير x بصورة لا نهائية.

(2-1) الحلقة التكرارية for

تستخدم الدالة for للتكرار خلال مجموعة من البيانات (تكرار محدد).

الصيغة العامة Syntax:

```
for variable in iterable:
    code block to be executed for each element.
```

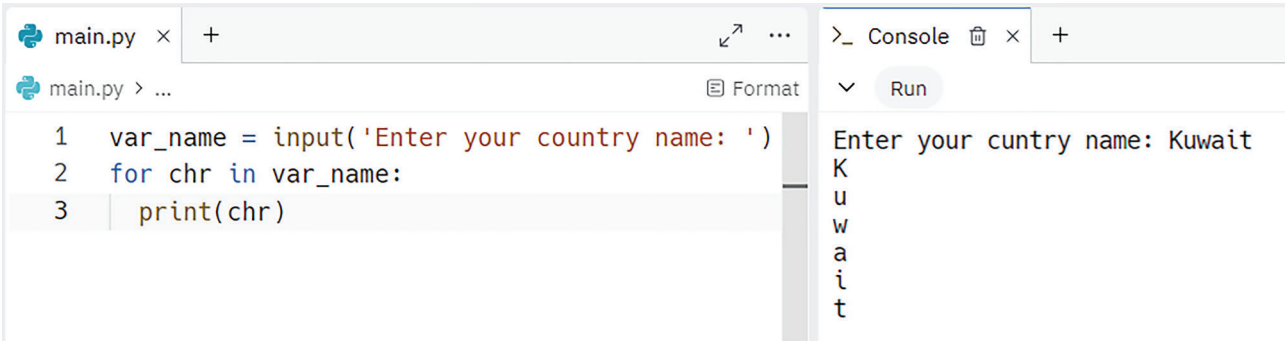


شكل (1-3) خريطة تدفق الصيغة العامة for

قابل للتكرار iterable: هي مجموعة محددة من العناصر مثل String – range().

مثال (1-2)

استخدام دالة input لإدخال الاسم ثم طباعة كل حرف من السلسلة النصية للمتغير في أسطر مستقلة .



يمكن استخدام الدالة ()range لإنشاء سلسلة من الأعداد الصحيحة

الصيغة	الوصف	مثال	عناصر السلسلة
range(stop)	الأعداد الصحيحة الموجبة من 1 وأقل من stop	range(10)	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
range(start,stop)	الأعداد الصحيحة من start وأقل من stop	range(3,8)	3, 4, 5, 6, 7
لمعلوماتك			
range(start, stop,step)	سلسلة من الأعداد الصحيحة من start وأقل من stop بزيادة step في مرة	range(1,10,2)	1, 3, 5, 7, 9

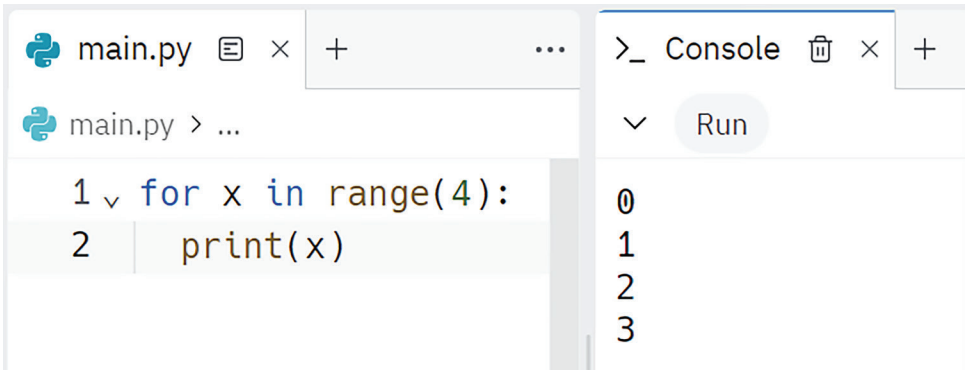
أكمل الجدول التالي:

المثال	السلسلة
range(7)	.....
range(2,10)	.....
range(-10,-1)	.....
range(10,2,-1)	.....
range(2,10,2)	.....

استخدام الحلقة for مع range

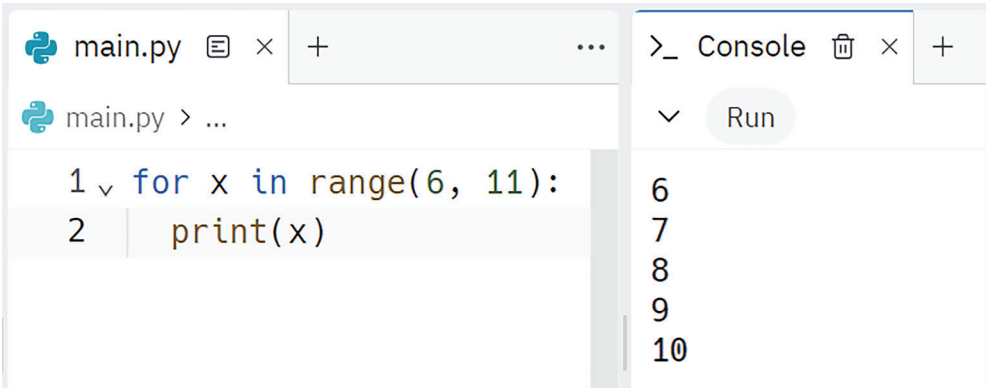
مثال (1-3)

استخدام الحلقة for لطباعة الأعداد من 0 إلى 3 على النحو التالي:



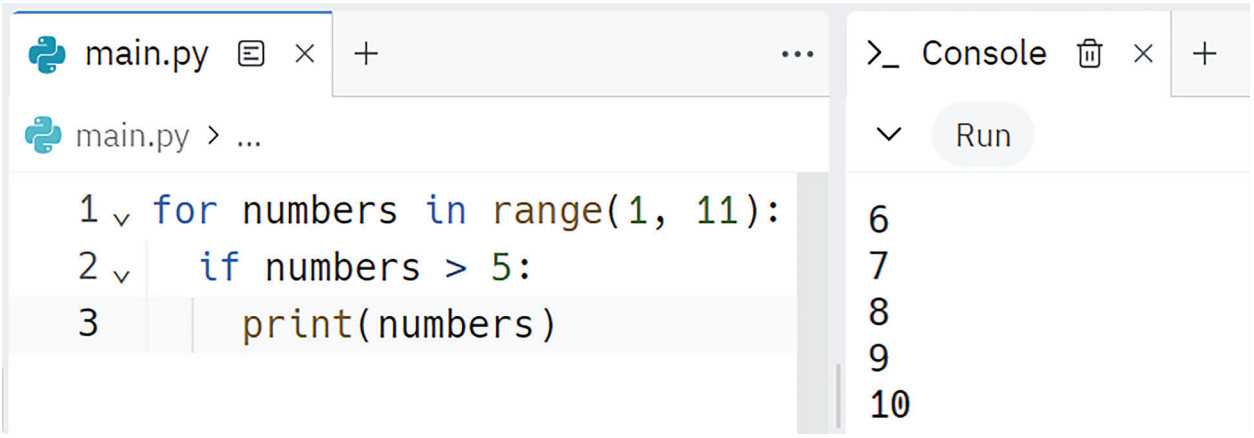
مثال (1-4)

استخدام الحلقة for لطباعة الأعداد من 6 إلى 10 على النحو التالي:



مثال (1-5)

لدينا مجموعة من الأرقام (من 1 إلى 10)، نريد استخدام التكرار لطباعة الأعداد أكبر من 5، ويمكننا القيام بذلك على النحو التالي:



مثال (1-6)

يطلب البرنامج إدخال كلمة المرور حتى يتم مطابقتها مع كلمة المرور الصحيحة، ثم يعرض رسالة «تم إدخال كلمة المرور بشكل صحيح».

```
main.py
main.py > ...
1  correctPassword = '1234'
2  password = ''
3  while (password != correctPassword):
4      password = input('Enter password: ')
5  print('Password Correct' )
```

إثرائي: طور البرنامج ليتم تحديد عدد محاولات الإدخال بثلاث محاولات فقط، يمكنك الاستعانة بمعلمك.

لمعلوماتك

(3-1) التكرارات المتداخلة nested loops

مما سبق تعلمت أن التكرار يقوم بتكرار مجموعة من التعليمات البرمجية، وهذه التعليمات يُمكن أن تحتوي بداخلها على تكرارات أخرى.

```
main.py
main.py
1  var_x = 0
2  var_y = 0
3  for r in range(2):
4      var_x += 1
5      for c in range(3):
6          var_y += 1
7  print(var_x)
8  print(var_y)
```

```
Console
Run
2
6
```

ناقش مع معلمك عدد مرات تخصيص المتغير var_x والمتغير var_y.

تستخدم التعليمة البرمجية break و continue، للتحكم في عملية التكرار.

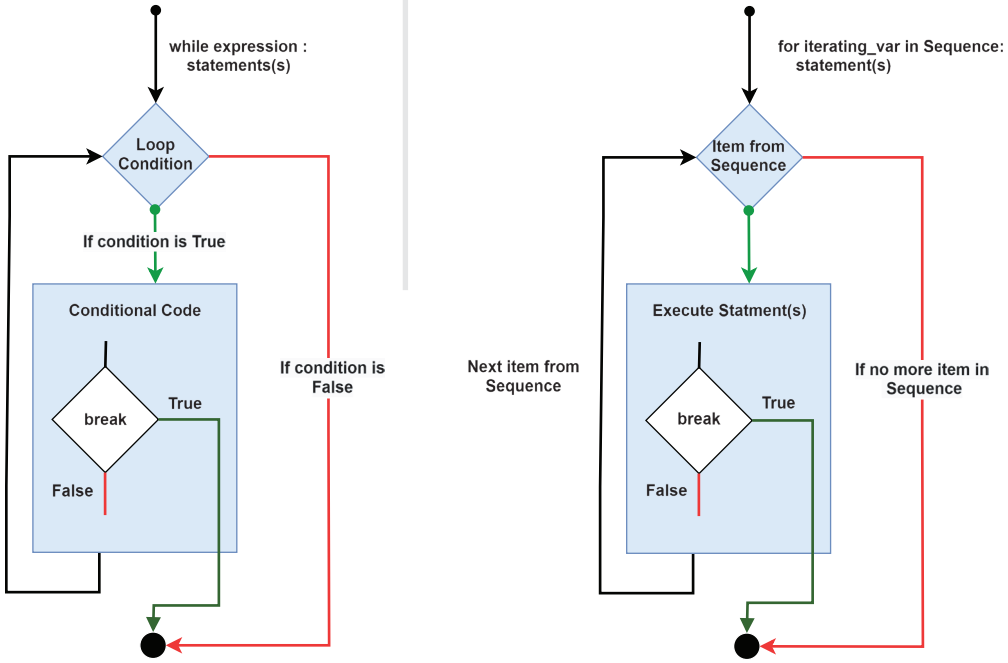
(4-1) التعليمة البرمجية break

تستخدم تعليمة break للخروج من الحلقة فورًا.

الصيغة العامة Syntax:

```
while condition:
    loop code
    if condition_to_break:
        break
    more loop code
```

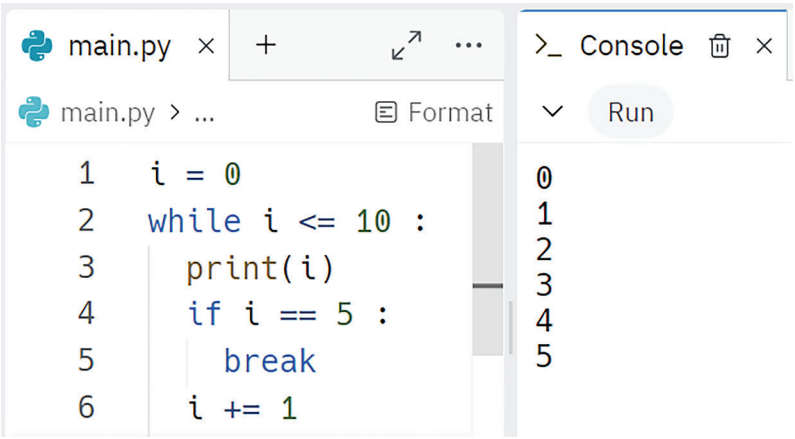
```
for variable in iterable:
    loop code
    if condition_to_break:
        break
    more loop code
```



شكل (4-1) خريطة تدفق الصيغة العامة break

مثال (1-7)

إذا كانت لدينا حلقة while تتكرر من 0 إلى 10، ونريد الخروج من الحلقة عندما تصل إلى 5، يمكننا استخدام تعليمات break على النحو التالي:



سيؤدي هذا النص البرمجي إلى طباعة الأرقام من 0 إلى 5. عند الوصول إلى 5، ستؤدي تعليمة break إلى خروج الحلقة، ولن يتم طباعة أي شيء بعد ذلك.

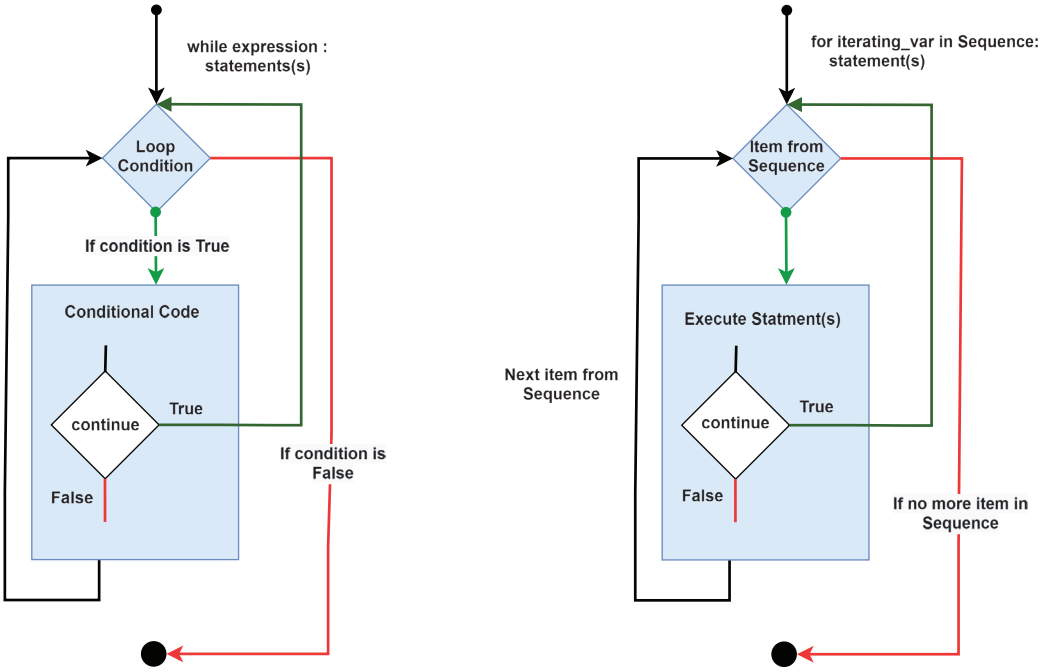
(5-1) التعليمة البرمجية continue

تستخدم دالة continue لتجاوز بقية النص البرمجي في الحلقة الحالية والانتقال إلى التكرار التالي.

الصيغة العامة Syntax:

```
while condition:
    loop code
    if condition_to_ continue:
        continue
    more loop code
```

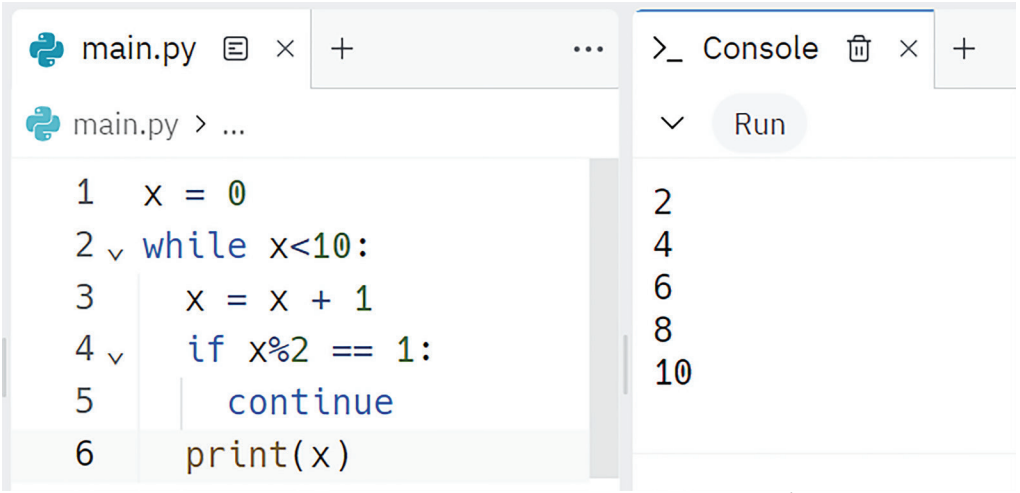
```
for variable in iterable:
    loop code
    if condition_to_continue:
continue
    more loop code
```



شكل (5-1) خريطة تدفق continue

مثال (1-8)

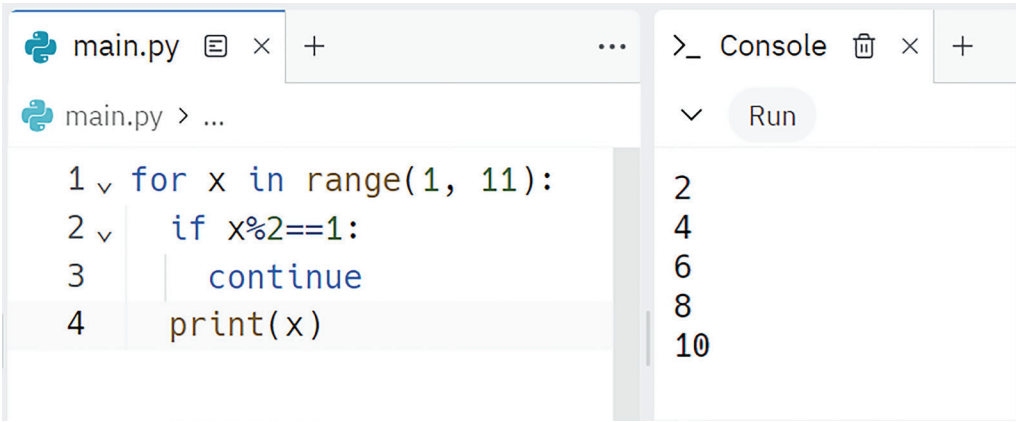
طباعة الأعداد الزوجية باستخدام while:



لاحظ: استخدام $x = x + 1$ أو $x += 1$، كلاهما يؤدي نفس الوظيفة، وهي زيادة قيمة المتغير بواحد.

مثال (1-9)

طباعة الأعداد الزوجية باستخدام for:

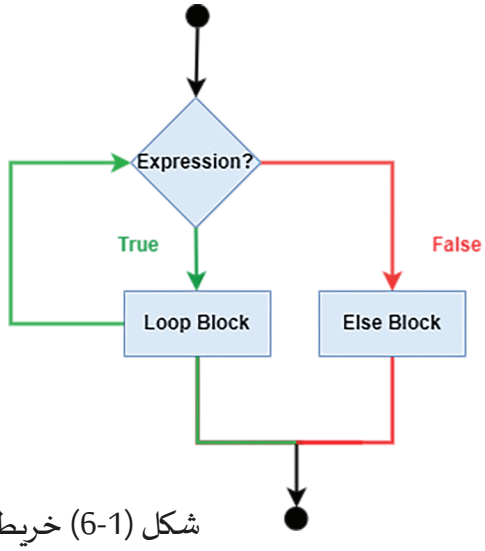


(6-1) دالة while ... else

يمكن تنفيذ الكتلة البرمجية else مع الحلقة التكرارية while.

الصيغة العامة Syntax:

```
while condition:
    loop body
else:
    code to be executed when loop finishes normally
```



شكل (1-6) خريطة تدفق الصيغة العامة **while ... else**

مثال (1-10)

برنامج بسيط يُظهررسالة (خطأ) عند محاولة إدخال المستخدم كلمة المرور3 مرات بصورة خاطئة.

main.py

```
n = 1
correct_password = 'a1234'
while n < 4:
    password = input('Enter password: ')
    if password == correct_password:
        break
    n = n + 1
else:
    print('You have entered the wrong
password 3 times.')

```

Console

```
Enter password: 123
Enter password: a123
Enter password: a234
You have entered the wrong password 3 times.
```

ورقة عمل (1-1)

برنامج لعبة تخمين عدد

مشكلة البرنامج:

إدخال أعداد صحيحة، والتأكد من مدى مطابقتها للعدد المُعرف في البرنامج.

الخوارزمية:

- الإعلان عن متغير يساوي العدد المطلوب تخمينه.
- استقبل عددًا صحيحًا من المستخدم.
- مطابقة العدد المُدخل مع العدد المحدد في الحالات التالية:

- إذا كان العدد أكبر يطبع رسالة العدد أكبر، ويستمر المستخدم في عملية الإدخال.

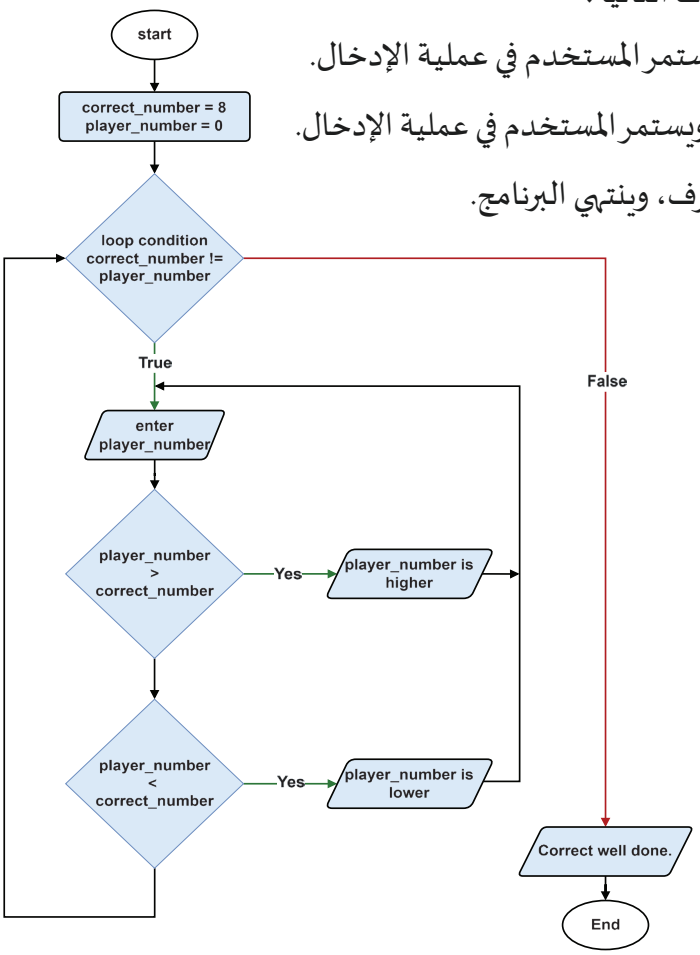
- إذا كان العدد أصغر يطبع رسالة العدد أصغر، ويستمر المستخدم في عملية الإدخال.

- طباعة رسالة العدد المُدخل مطابق للعدد المُعرف، وينتهي البرنامج.

المطلوب:

خريطة التدفق

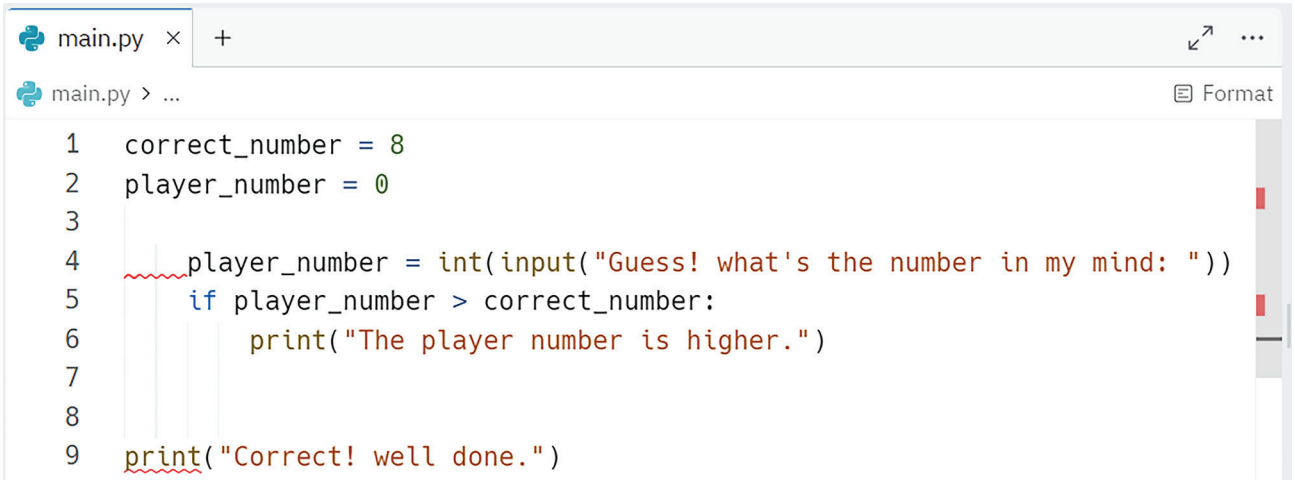
- ادرس خريطة التدفق المقابلة.



شكل (1-7) خريطة تدفق برنامج تخمين عدد صحيح

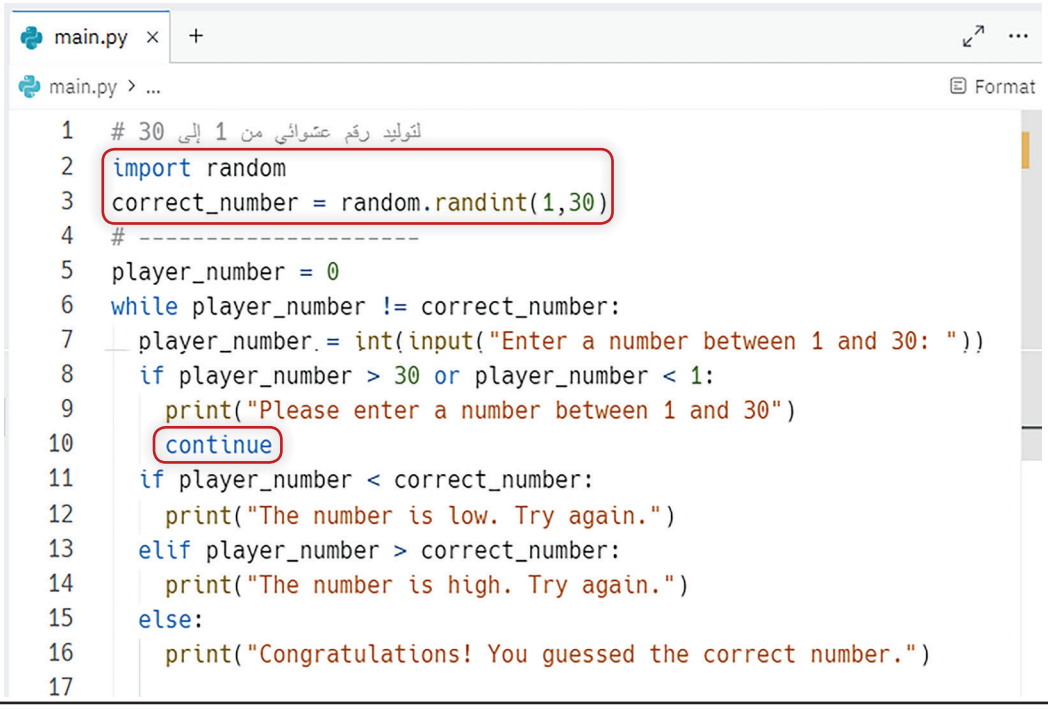
البرنامج

- افتح الملف correct_number.py.



- أكمل التعليمات البرمجية اللازمة ليعمل البرنامج باستمرار حتى يتطابق العدد المُدخل مع العدد المُعرف.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

إثرائي: ناقش مع معلمك التعليمات البرمجية لتطوير البرنامج السابق لتوليد رقم عشوائي:



ورقة عمل (1-2)

برنامج جدول الضرب

مشكلة البرنامج:

برنامج يطبع جدول الضرب من 1 إلى 12 لعدد محدد.

الخوارزمية:

- استقبل من المستخدم العدد المطلوب طباعة جدول الضرب الخاص به.
- **إذا** كان العدد أصغر من أو يساوي 12.

- إنشاء **حلقة تكرارية** لطباعة ضرب العدد المُدخل في الأعداد من 1 إلى 12.

- استخدام معادلة الناتج = العدد المُدخل * قيمة متغير التكرار.

- **إذا** كان العدد أكبر من 12، يطبع رسالة تنبيه.

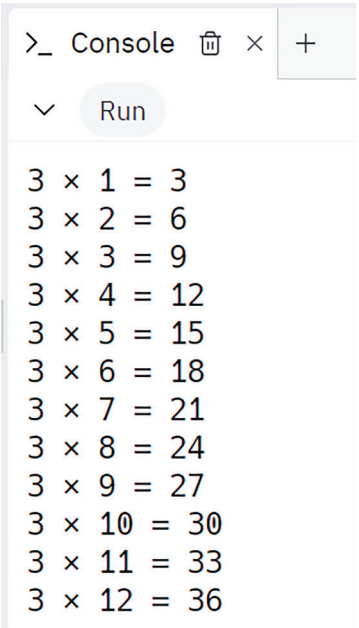
المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- أنشيء ملف Python باسم Multiplication_table.py.
- اكتب التعليمات البرمجية اللازمة لطباعة جدول الضرب لعدد محدد، (مثال العدد المحدد 3).
- اختبر البرنامج، وتأكد من عدم وجود أخطاء.



خريطة التدفق

برمجة Python

السلاسل النصية Strings

نواتج التعلم

- تعريف السلاسل النصية وخصائصها.
- إجراء العمليات الأساسية على السلاسل النصية.
- التعامل مع دوال السلاسل النصية.
- تحويل السلاسل النصية إلى أنواع أخرى من البيانات.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

السلاسل النصية Strings

السلاسل النصية String

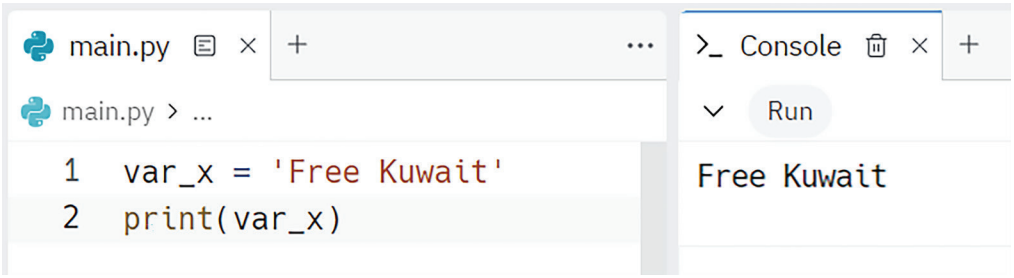
هي مجموعة من الأحرف والأرقام والرموز تُميز بعلامات اقتباس، تخزن السلاسل في متغيرات يتم التعامل معها كقيمة نصية.

(1) عمليات السلاسل النصية

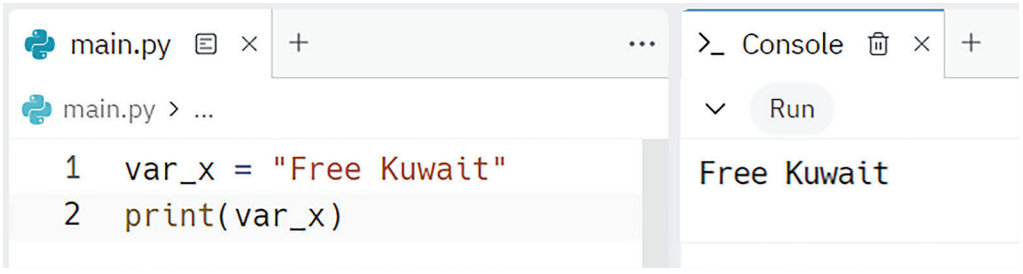
(1-1) تعريف وتخصيص وطباعة السلاسل النصية:

يتم تعريف وتخصيص وطباعة السلاسل النصية بعلامات اقتباس فردية ' ' أو مزدوجة " ".

مثال (1-1) إنشاء سلسلة نصية باستخدام علامتي اقتباس فردية.



مثال (1-2) إنشاء سلسلة نصية باستخدام علامتي اقتباس مزدوجة.

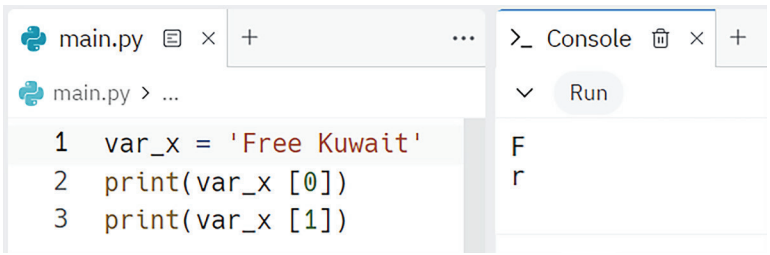


(2-1) الفهرسة[ ]Indexing: الوصول إلى موقع الأحرف في سلسلة نصية[x].

- تبدأ عملية الفهرسة من رقم صفر.

Index Number	0	1	2	3	4	5	6	7	8	9	10
String	F	r	e	e		K	u	w	a	i	t

مثال (1-3)

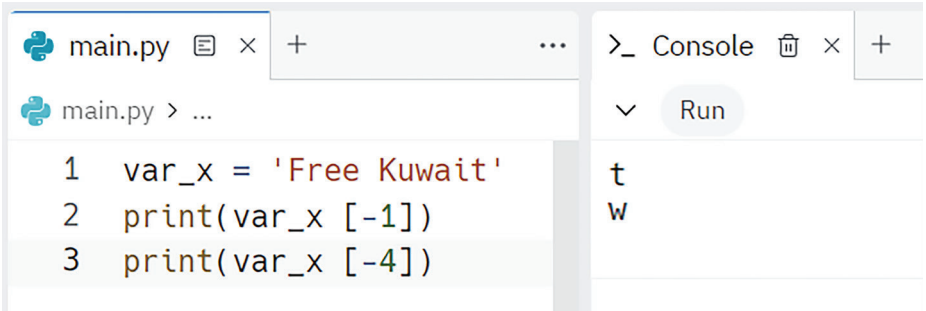


لمعلوماتك

تبدأ عملية الفهرسة بترتيب عكسي من نهاية السلسلة النصية «برقم 1- والرقم الذي يسبقه 2- وهكذا».

Index Number	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1
String	F	r	e	e		K	u	w	a	i	t

مثال (1-4)

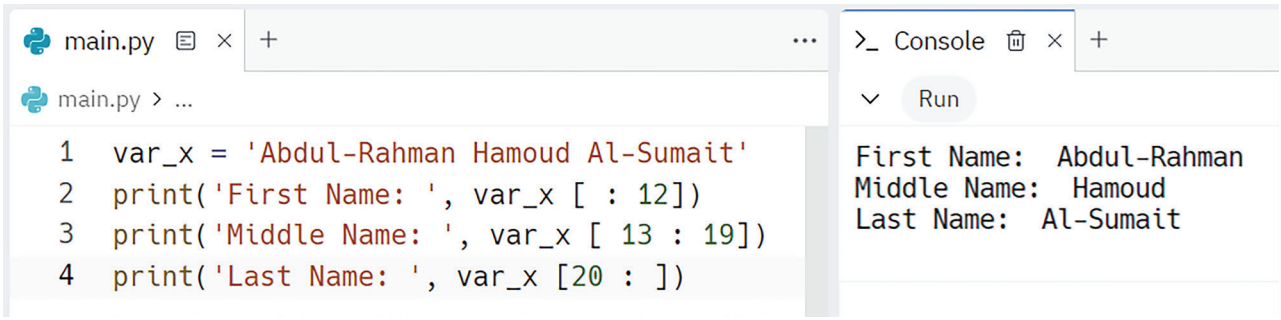


(3-1) القطع Slicing:

يمكن الحصول على جزء من السلسلة النصية بأرقام الفهارس [x : y].

الوصف	الصيغة
تُعيد النص بداية من الموقع x وحتى الموقع ما قبل y	str[x : y]
تُعيد النص بداية من الموقع 0 وحتى الموقع ما قبل y	str[: y]
تُعيد النص بداية من الموقع x وحتى نهاية السلسة النصية	str[x :]

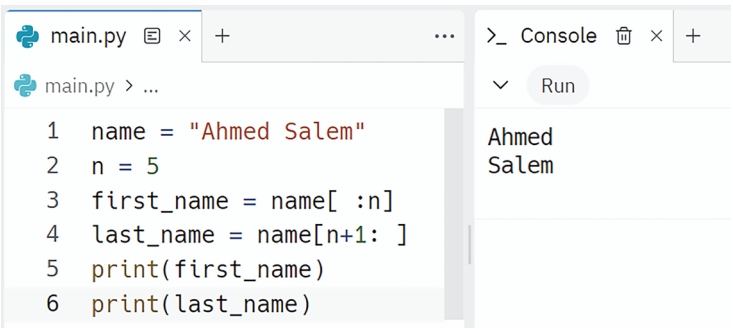
مثال (1-5)



مثال (1-6)

يمكن تقسيم نص عند موقع محدد بحيث نحصل على ما قبل الموقع وما بعده:

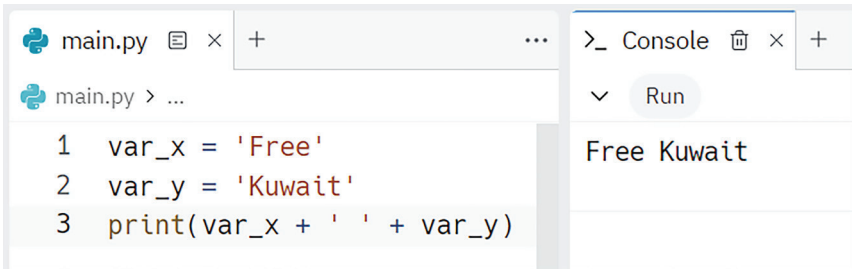
على سبيل المثال، يُمكن تقسيم اسم الطالب إلى (الاسم الأول، واسم العائلة) بحيث نحصل على الاسم الأول واسم العائلة بشكل منفصل (موقع التقسيم n=5).



(4-1) جمع السلاسل النصية Concatenation

يستخدم معامل عملية الجمع (+) لإضافة سلسلتين نصيتين لإنشاء سلسلة نصية جديدة.

مثال (1-7)



لاحظ:

يُمكن استخدام علامات التنصيص، لإضافة مسافة بين المتغيرين.

مثال (1-8)

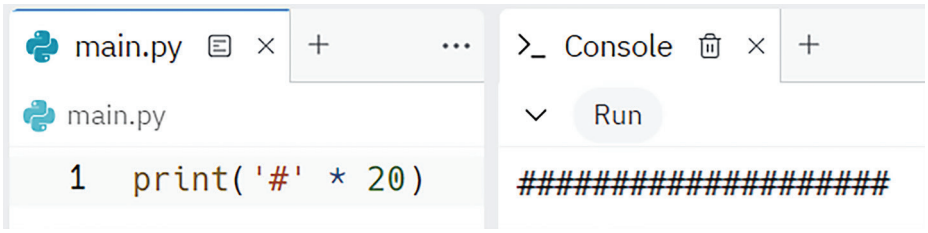
لاحظ:

تم إضافة المسافة بعد كلمة :Day

(5-1) تكرار السلاسل النصية:

تكرار سلسلة نصية لأي عدد باستخدام معامل الضرب (*).

مثال (1-9)

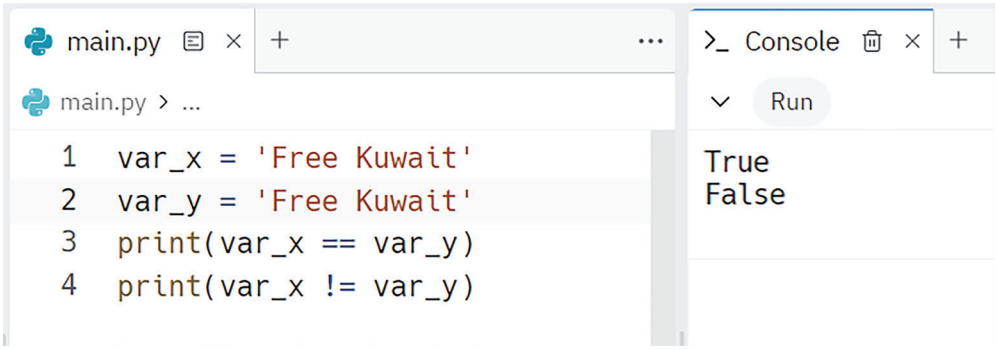


لمعلوماتك

(6-1) المقارنة Comparison:

مقارنة السلاسل النصية باستخدام معامل المقارنة == ، =! .

مثال (1-10)

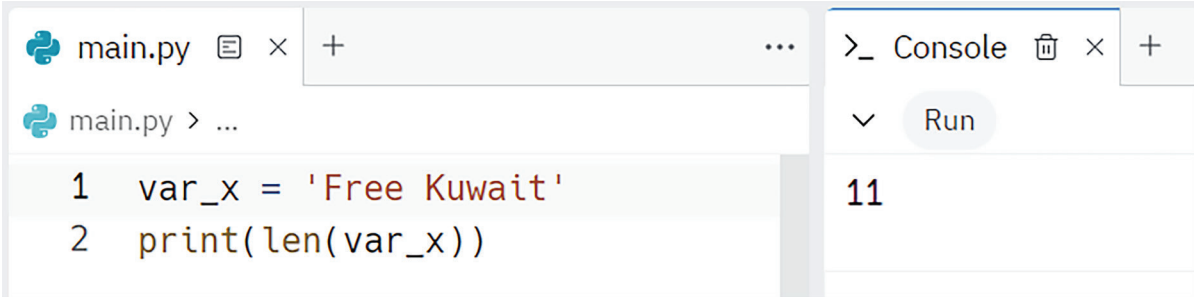


(2) دوال السلاسل النصية

(1-2) len( ):

تُستخدم للحصول على طول السلسلة النصية.

مثال (1-11)



(2-2) capitalize( ):

تُستخدم لتحويل الحرف الأول إلى حرف كبير. string.capitaliz()

مثال (1-12)

```
main.py
1  var_x = 'hello, cybersecurity!'
2  print(var_x.capitalize())
```

```
Console
Run
Hello, cybersecurity!
```

(3-2) upper( ):

تُحول جميع الأحرف في السلسلة النصية إلى أحرف كبيرة. string.upper()

مثال (1-13)

```
main.py
1  var_x = 'Hello, Cybersecurity!'
2  print(var_x.upper())
```

```
Console
Run
HELLO, CYBERSECURITY!
```

(4-2) lower( ):

تُحول جميع الأحرف في السلسلة النصية إلى أحرف صغيرة. string.lower()

مثال (1-14)

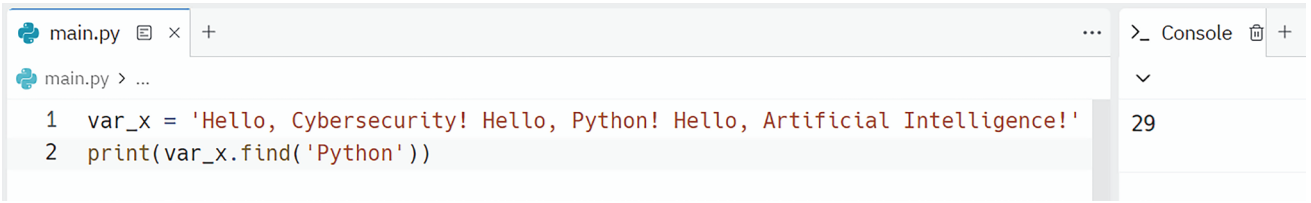
```
main.py
1  var_x = 'COMPUTER SCIENCE'
2  print(var_x.lower())
```

```
Console
Run
computer science
```

find( ) (5-2):

تستخدم لإرجاع رقم فهرس لأول ظهور لسلسلة فرعية في السلسلة المحددة. string.find(sub)

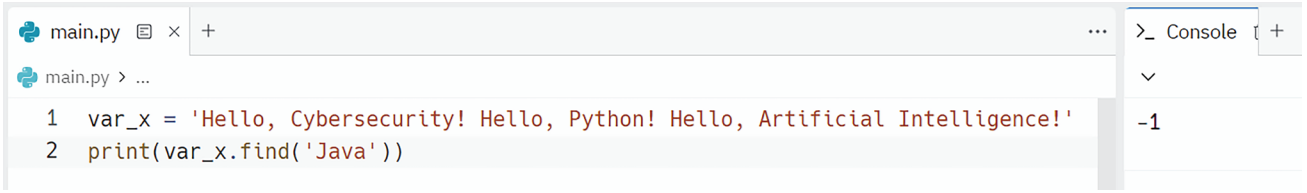
مثال (1-15)



لاحظ:

إذا لم يتم العثور على النص، فستُرجع دالة find() العدد -1.

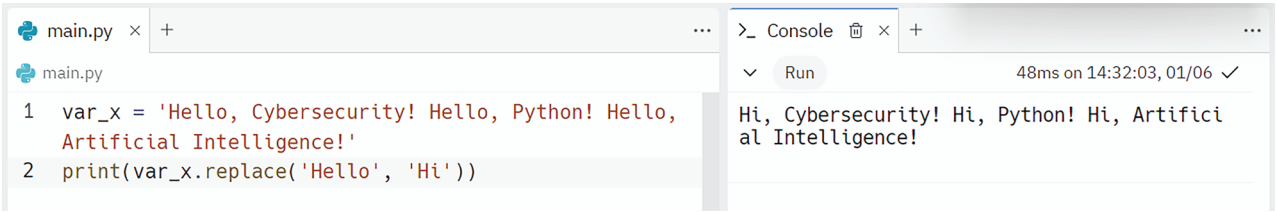
مثال (1-16)



replace( ) (6-2):

دالة تستخدم لاستبدال نص بآخر في السلسلة النصية. string.replace(oldvalue, newvalue)

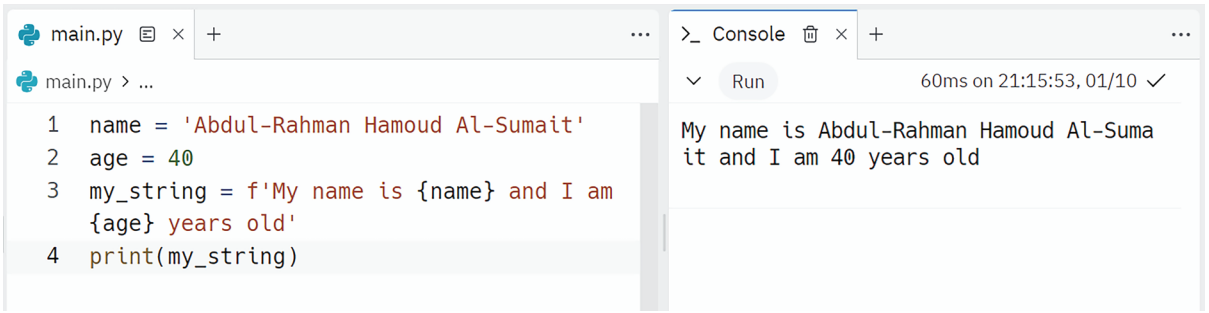
مثال (1-17)



(7-2) التعليمة f-string:

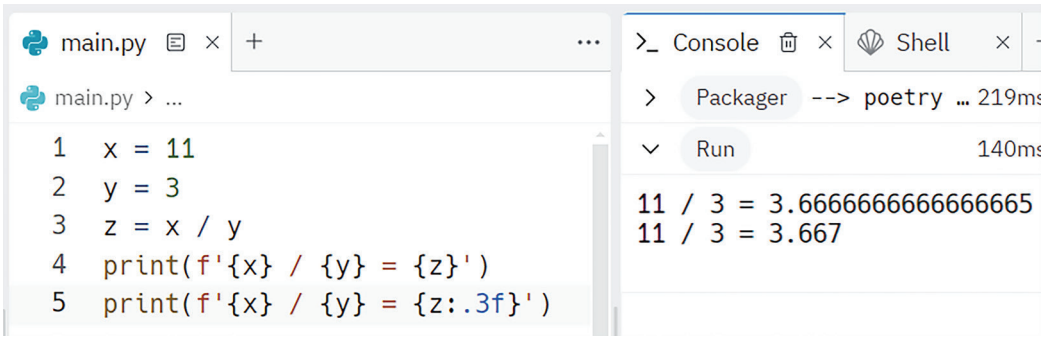
تستخدم لإنشاء تعبيرات برمجية (strings) تتضمن قيمًا متغيرة لتضمين قيم من أنواع بيانات مختلفة.

مثال (1-18)



لمعلوماتك

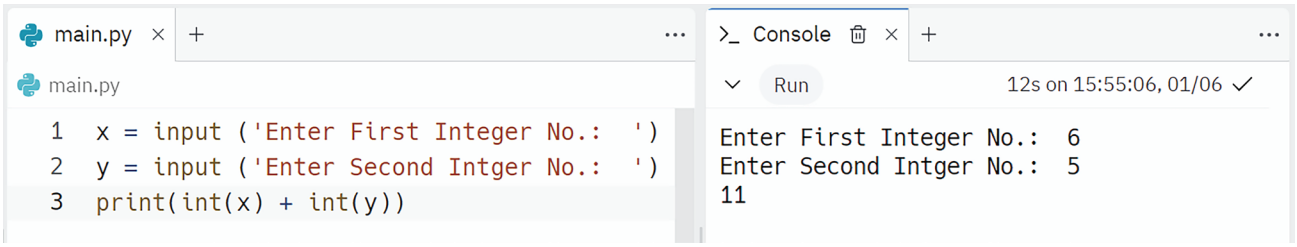
يمكن التحكم في تنسيق الأرقام العشرية.



(8-2) دالة ( )int: تُستخدم لتحويل السلاسل النصية إلى أعداد صحيحة:

درست أن دالة ()input تُرجع البيانات المُدخلة كسلسلة نصية string، ولتحويلها إلى intger تستخدم دالة ()int.

مثال (1-19)

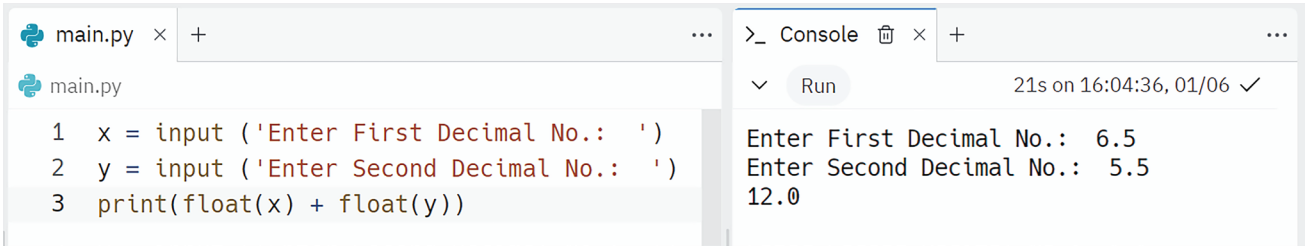


لاحظ: مثال جمع السلاسل النصية.

(9-2) دالة ( )float:

تُستخدم لتحويل السلاسل النصية (أرقام numeric)، أو الأعداد الصحيحة إلى أعداد عشرية.

مثال (1-20)



(10-2) دالة ( )count:

تُستخدم لإيجاد عدد مرات ظهور سلسلة نصية.

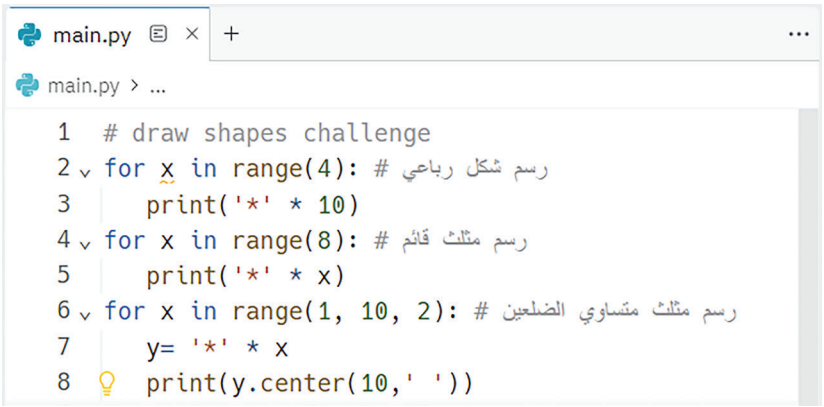
(11-2) دالة ( )center:

تُعيد أحرف السلسلة مزاحة نحو الوسط ضمن سلسلة ذات طول محدد.

مثال (1-21)

مثال تطبيقي (1-22)

فكرة البرنامج: استخدام تكرار السلاسل النصية، ودالة center، لرسم الأشكال الهندسية من خلال الأحرف أو الرموز الخاصة، افتح الملف draw shapes string.py، شغل البرنامج، وناقش مع معلمك وزملائك آلية بناء الجمل البرمجية.



غيرفي عدد مرات التكرارومعامل العرض للدالة Center، ثم لاحظ الفرق عند تشغيل البرنامج.

ناقش التعليمات البرمجية التالية، لرسم شكل المعين.

```
9  for x in range(1, 9, 2): # رسم معين
10     y = '*' * x
11     print(y.center(10, ' '))
12 for x in range(9, 0, -2):
13     y = '*' * x
14     print(y.center(10, ' '))
```

ورقة عمل (1-1)

برنامج عكس السلسلة النصية

مشكلة البرنامج

إنشاء سلسة نصية، بحيث تبدأ من النهاية وحتى البداية.

الخوارزمية

- عرف متغير بقيمة ابتدائية فارغة.
- استقبل من المستخدم العبارة النصية المطلوب عكس أحرفها.
- أنشئ حلقة تكرارية وفقًا لعدد أحرف العبارة النصية المُدخلة.
- استخدم معادلة تجميع النصوص التالية:

المتغير الخاص بتجميع أحرف السلسلة النصية =

حرف السلسلة وفق التكرار + المتغير الخاص بتجميع أحرف السلسلة النصية.

- اطبع البرنامج المتغير الخاص بتجميع أحرف السلسلة النصية.

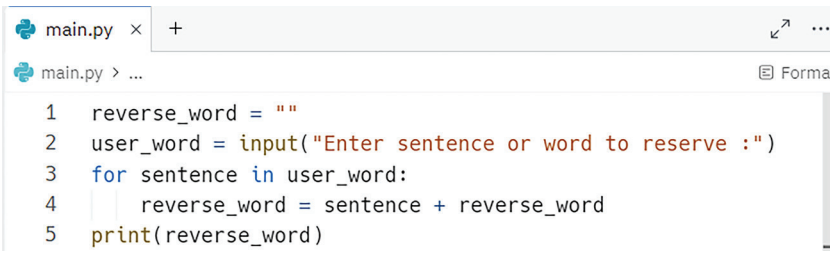
المطلوب:

خريطة التدفق

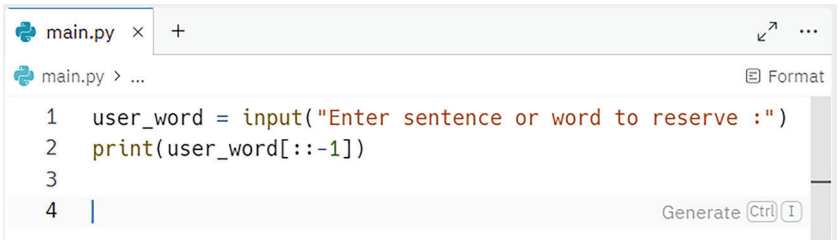
- ارسم خريطة التدفق المناسبة.

خريطة التدفق

- ادرس التعليمات البرمجية المقابلة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.



• ويمكن حل المشكلة، كما هو موضح في التعليمات البرمجية التالية:



خريطة التدفق

ورقة عمل إثرائية (1-2)

برنامج لغز التخمين

مشكلة البرنامج:

يستنتج المستخدم الكلمة الصحيحة للغز، وفق رسالة توضيحية تظهر له من البرنامج.

الخوارزمية:

1- طباعة رسالة توضيحية حول اللغز.

2-الإعلان عن متغير يمثل كلمة اللغز.

3-تظهر رسالة توضح عدد أحرف إجابة اللغز ممثلة بالرمز (-)، مضروبة بعدد أحرف الكلمة.

4-إنشاء حلقة تكرارية، تستمر حتى تتوافق إجابة المستخدم مع الإجابة الصحيحة.

- استقبال من المستخدم إحدى حروف كلمة اللغز.
- إذا كان الحرف المُدخل يطابق إحدى أحرف كلمة اللغز
- تحديد موضع (Indexing) الحرف المُدخل من الكلمة الصحيحة للغز.

تعديل قيمة المتغير إ(جابة المستخدم) ليصبح كالتالي:

إجابة المستخدم = الأحرف من بداية قيمة متغير إجابة المستخدم، وحتى موقع الحرف الصحيح + الحرف الصحيح المُدخل + الأحرف من بعد موقع الحرف الصحيح المُدخل حتى نهاية قيمة متغير إجابة المستخدم.

- طباعة رسالة بالأحرف الصحيحة المُدخلة، والرمز (-) بالأحرف الناقصة.

5- إظهار رسالة تشجيعية للمستخدم.

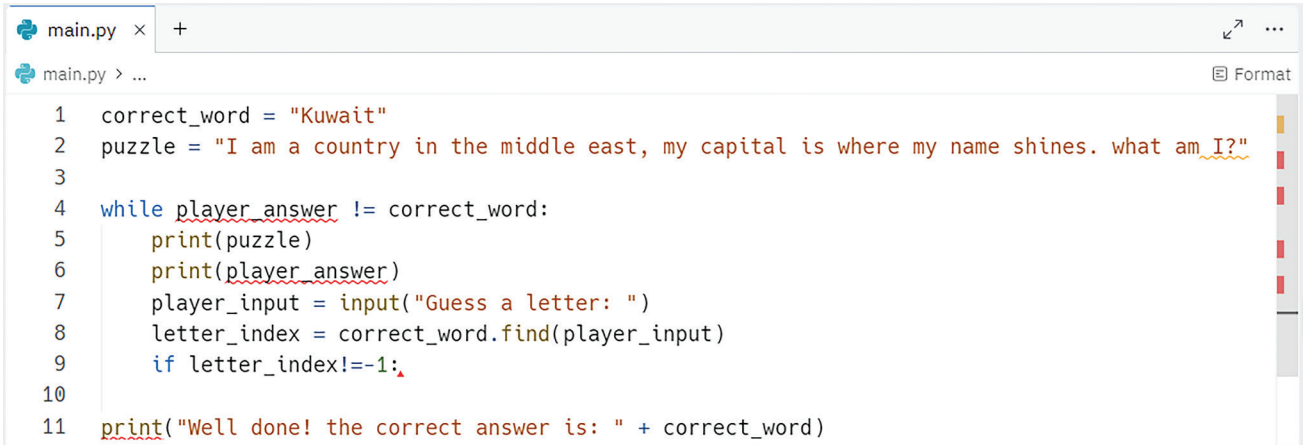
المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- افتح الملف correct_word.py، ثم صحح الأخطاء واستكمل التعليمات ليعمل البرنامج بشكل صحيح.
- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.



ناقش مع معلمك تعديل التعليمات البرمجية لتنفيذ التالي:

- أن تكون كلمة اللغز بها أحرف مكررة.

خريطة التد

برمجة Python

القوائم Lists

نواتج التعلم

- التعرف على القوائم وكيفية إنشائها.
- التعامل مع العمليات الخاصة بالقوائم.
- استخدام الدوال الخاصة بالقوائم في التعليمات البرمجية.

يمثل رمز الاستجابة السريعة QR رابط لملفات أوراق العمل، ومصادر التعلم.

القوائم Lists

البيانات غير الأولية Non-Primitive Data Structures

تم عرض البيانات الأولية والبيانات غير الأولية، وفي هذا الدرس نستعرض البيانات غير الأولية الخطية Data Liner Structures:

- القوائم Lists
- الصفوف Tuples

خصائص البيانات الخطية Linear Data Structures

تتوافر هذه الخصائص في بعض أنواع البيانات الخطية.

- الترتيب Ordering: ترتيب العناصر مع كل عملية تشغيل Run.
- قابلة للتغيير Mutability: إمكانية تغيير عناصرها.
- الفهرسة Indexing: كل عنصر في القائمة يقابله رقم صحيح يمثل فهرس ذلك العنصر (فهرس العنصر الأول في القائمة= 0).
- العناصر الفريدة Unique Elements: إمكانية تكرار قيم البيانات.
- أنواع مختلفة من البيانات Different of Data Types: إمكانية توافر أنواع مختلفة من البيانات.

(1) القوائم Lists

تمثل مجموعة مرتبة من البيانات تحتوي على أي نوع من البيانات، (الأعداد الصحيحة والعشرية والسلاسل النصية والكائنات، ...).

نوع البيانات Data Type	قابلة للتغيير Mutable	العناصر مرتبة Ordered	الفهرسة Indexing	العناصر الفريدة Unique Elements	أنواع مختلفة من البيانات Different of Data Types	الصيغة Syntax
List	نعم	نعم	نعم	لا	نعم	[, ,] Square Brackets

(1-1) إنشاء القوائم

يتم إنشاء القوائم باستخدام الأقواس المربعة [ ]، ويفصل بين عناصرها علامة الفاصلة Comma ,.

مثال (1-1)

يمكن إنشاء قائمة من عدة أنواع البيانات مثل الأعداد الصحيحة والعشرية والسلاسل النصية.

```
main.py
1  numbers = [1,2,3,4,5] # قائمة من الأعداد الصحيحة
2  decimals = [1.0,2.5,3.14] # قائمة من الأرقام العشرية
3  strings = ['Hello','world!','Iam','a Student'] # قائمة من السلاسل النصية
4  multi = ['Hello','world!','Iam',16,'years','old'] # قائمة من أنواع مختلفة من البيانات
```

مثال (1-2)

يمكن إنشاء قائمة فارغة.

```
main.py
1  list_1 = [ ]
```

لمعلوماتك

مثال (1-2)

يمكن إنشاء قائمة من عدة قوائم مختلفة.

```
main.py
1  name_degree = [['Fahed', 'Ahmed', 'Ali'], [13, 15, 20]]
```

(2-1) العمليات على القوائم

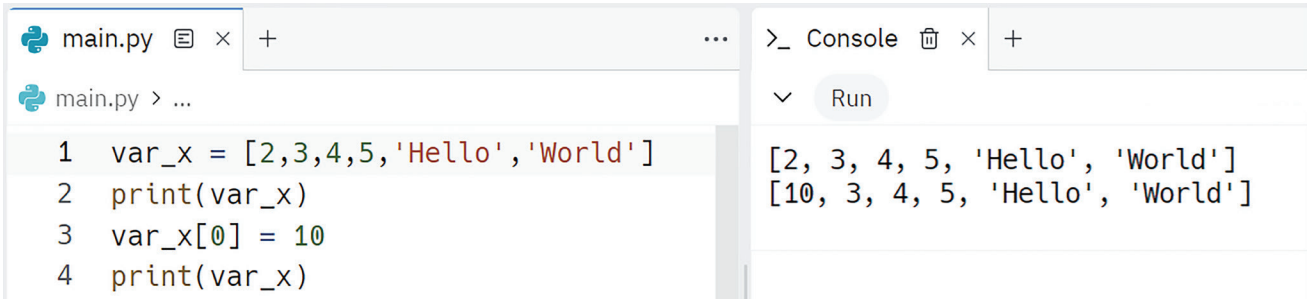
(1-2-1) فهرسة Indexing عناصر القائمة List

list_name[index]

مثال (1-3)

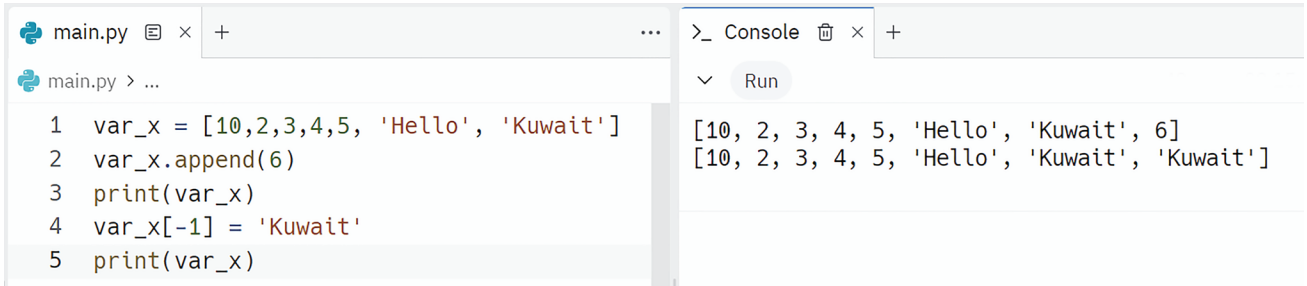
(2-2-1) تعديل عناصر القائمة List

مثال (1-4)



(3-2-1) إضافة وتحرير عناصر للقائمة List

مثال (1-5)



(4-2-1) حذف عناصر من القائمة List

مثال (1-6)

حذف عنصر محدد حسب القيمة من القائمة.

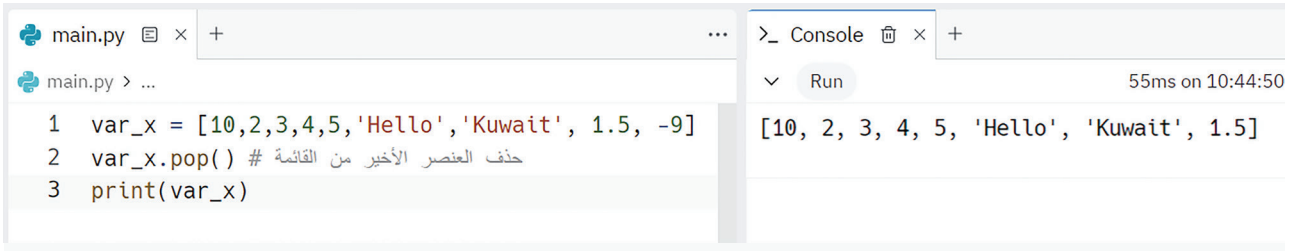
مثال (1-7)

حذف عنصر محدد حسب رقم indexing من القائمة.

لاحظ الفرق بين استخدام remove و pop.

مثال (1-8)

حذف العنصر الأخير من القائمة.



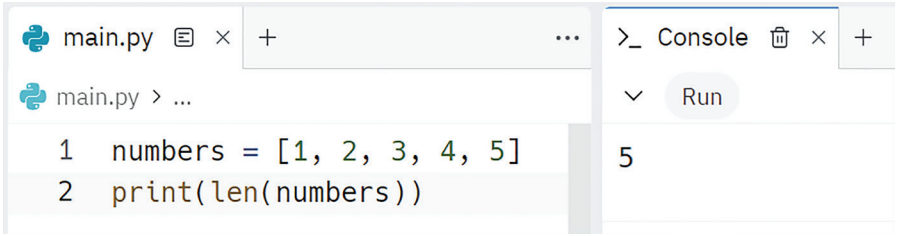
لاحظ:

يمكن حذف جميع عناصر القائمة باستخدام دالة ( )clear.

(1-2-5) الحصول على طول القائمة - عدد عناصرها ( )len.

مثال (1-9)

مثال (1-10)



(1-2-6) التأكد من وجود عنصر في القائمة if in:

مثال (1-11)

(7-2-1) تعبئة قائمة من المستخدم.

مثال (1-12)

ناقش مع معلمك فكرة هذا البرنامج ولماذا تم استخدام دالة ()pop.

```
main.py
main.py > ...
Format
1  students_list = [] # قائمة فارغة
2  students_name = 'stop' # القيمة الابتدائية للمتغير
3  while students_name != '': # عملية التكرار
4      students_name = input('Enter student name or press Enter key to exit loop: ') # إدخال الأسماء
5      students_list.append (students_name) # إضافة الاسم إلى القائمة
6  # students_list.pop( ) # حذف آخر عنصر (اسم) تم إدخاله
7  print(students_list) # طباعة القائمة
```

(8-2-1) التعامل مع عناصر القائمة من خلال عملية التكرار.

مثال (1-13)

طباعة عناصر القائمة.

```
main.py
main.py > ...
1  animals = ['cat', 'tiger', 'lion']
2  for animal in animals:
3      print(animal)
```

```
Console
Run
cat
tiger
lion
```

مثال (1-14)

لإرجاع نطاقٍ من الأعداد حسب طول القائمة.

```
main.py
main.py > ...
1  animals = ['cat', 'tiger', 'lion']
2  for n in range(len(animals)):
3      print(animals[n])
```

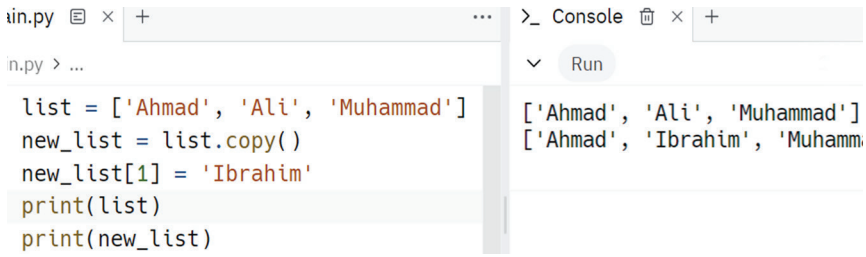
```
Console
Run
cat
tiger
lion
```

لمعلوماتك

(1-2-9) نسخ قائمة:

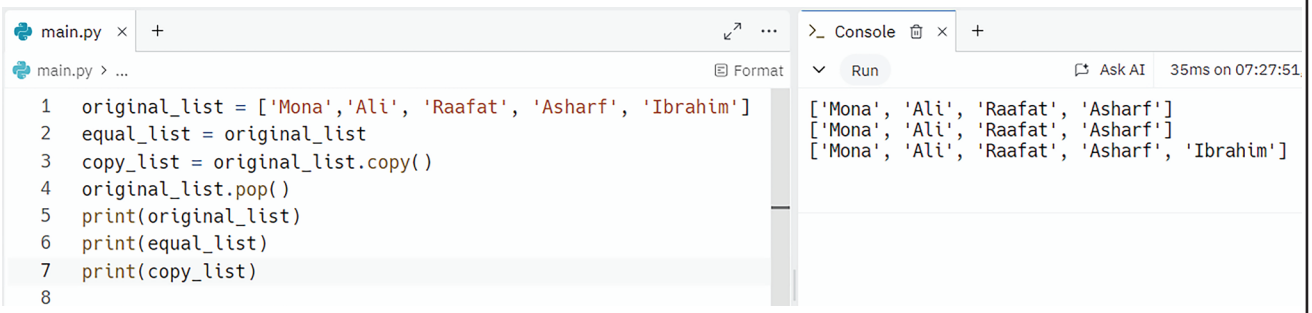
يمكن نسخ القائمة باستخدام دالة ()copy، مع ملاحظة أن التغيير في محتوى القائمة الجديدة لايغير في محتوى القائمة الأصلية.

مثال (1-15)



يمكن نسخ القائمة باستخدام دالة ()copy، مع ملاحظة أن التغيير في محتوى القائمة الجديدة لايغير في محتوى القائمة الأصلية.

ناقش مع معلمك التعليمات البرمجية التالية:

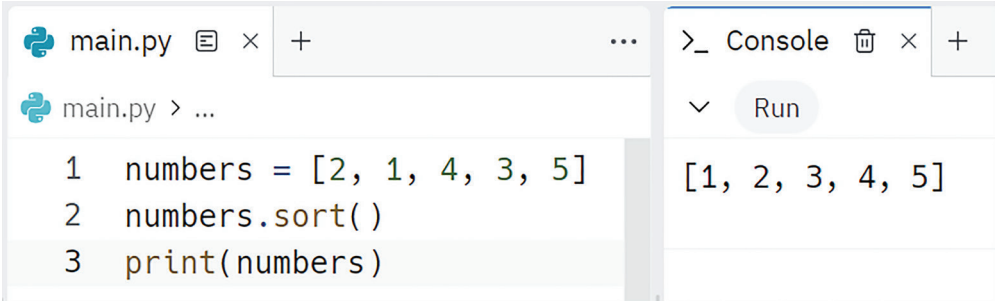


(10-2-1) عمليات الفرز على القوائم

تستخدم لترتيب العناصر في القائمة. فيما يلي بعض الأمثلة على عمليات الفرز على القوائم.

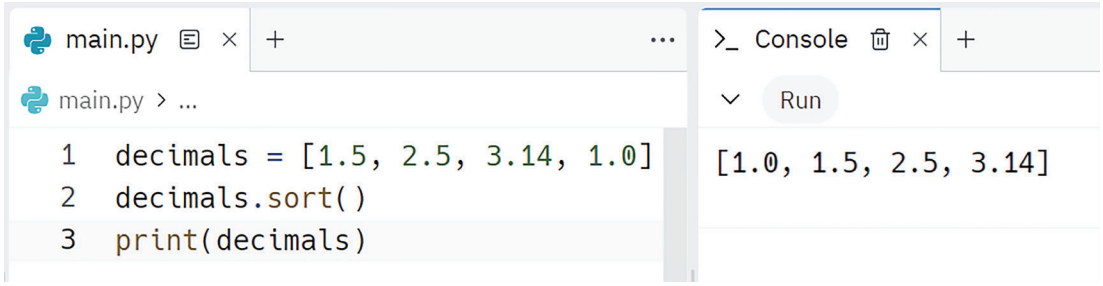
مثال (1-16)

فرز القائمة حسب الأعداد الصحيحة



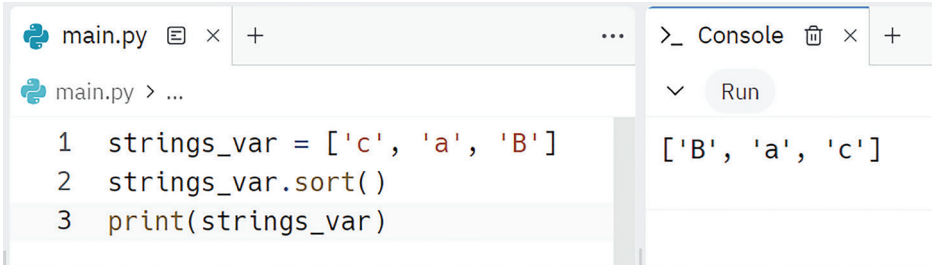
مثال (1-17)

فرز القائمة حسب الأعداد العشرية



مثال (1-18)

فرز القائمة حسب السلاسل النصية



لاحظ:

- ظهور حرف B في بداية القائمة حيث يتم ترتيب الحروف Capital ثم ترتيب الحروف Small.
- ترتب البيانات فقط في حال كان نوع البيانات داخل القائمة من نفس النوع.

لمعلوماتك

(11-2-1) split( ):

تقسيم السلسلة النصية إلى قائمة من السلاسل النصية. string.split(separator)

مثال (1-19)

تقسيم السلسلة النصية إلى قائمة من السلاسل النصية على أساس علامة التنصيص الفردية.

```
main.py
1  var_x = 'Cybersecurity Python Artificial Intelligence'
2  print(var_x.split(' '))
```

```
Console
Run
['Cybersecurity', 'Python', 'Artificial', 'Intelligence']
```

يمكن استخدام الخاصية maxsplit لتقسيم القائمة لسلاسل نصية بعدد محدد.

string.split(separator, maxsplit)

مثال (1-20)

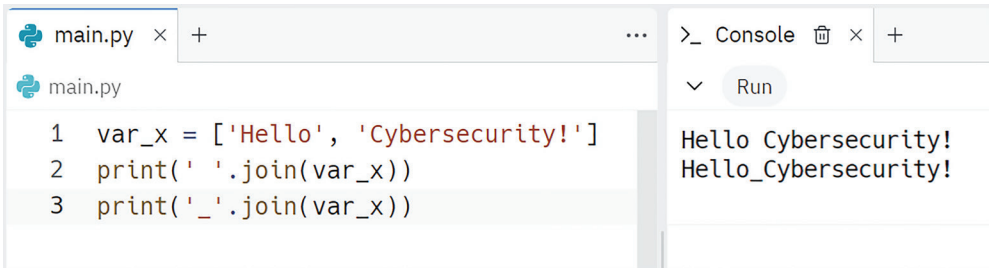
```
main.py
1  string = "Learn Python with Computer Technical Supervision"
2  split_string = string.split(" ", 3)
3  print(split_string)
```

```
Console
Run
['Learn', 'Python', 'with', 'Computer Technical Supervision']
```

(12-2-1) دالة ( )join:

تجمع قائمة من السلاسل النصية في سلسلة نصية واحدة. string.join(iterable)

مثال (1-21)



بعض الدوال التابعة للكائن list

list.extend() إضافة مجموعة من العناصر إلى نهاية القائمة.

list.insert() إضافة عنصر إلى القائمة في الموقع الذي يحدّده المستخدم.

list.index() تحديد موقع العنصر ضمن القائمة.

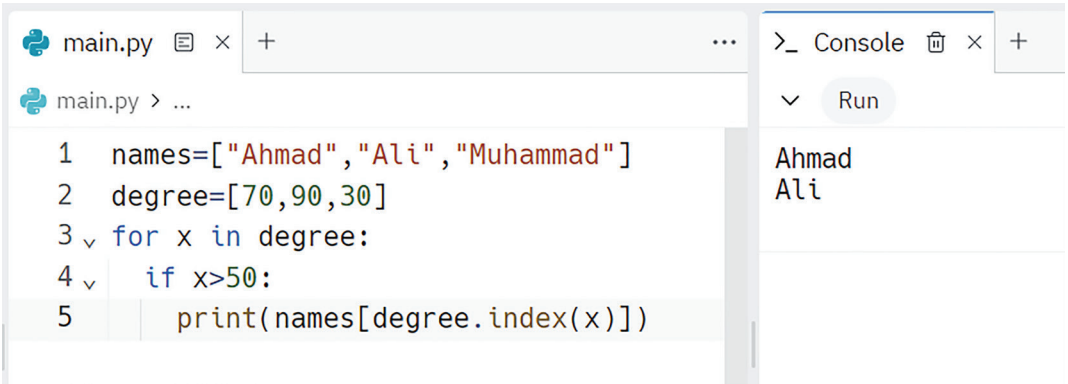
list.count() تحديد عدد مرات تكرار العنصر الذي يحدّده المستخدم في القائمة.

list.reverse() قلب ترتيب عناصر القائمة في مكانها.

أفكار أمثلة إضافية

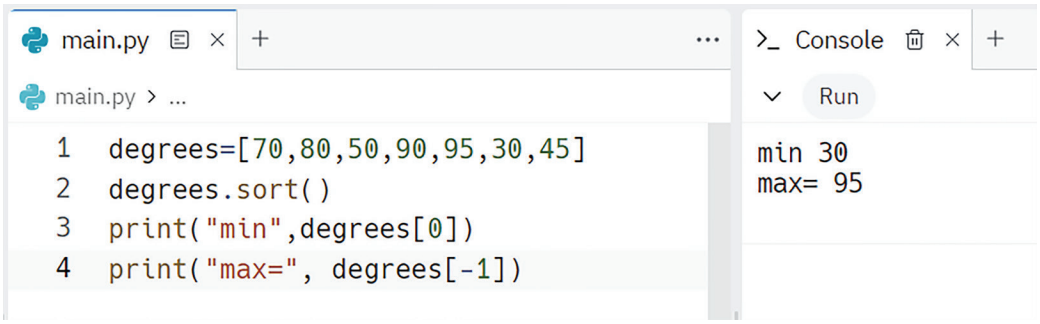
مثال (1-22)

لديك قائمة تضم أسماء الطلاب وقائمة تضم درجات الطلاب المطلوب طباعة أسماء الطلاب الناجحين (الحاصلين على درجات أكبر من 50).



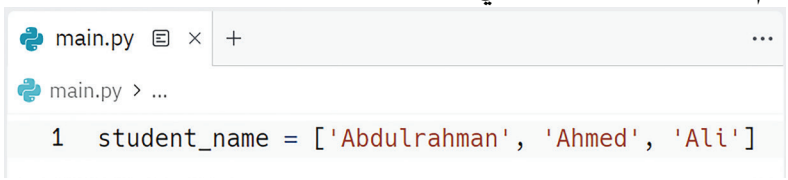
مثال (1-23)

لديك قائمة تتضمن درجات الطلاب، مطلوب طباعة أقل / أكبر درجة بالقائمة.



مثال (1-24)

لديك قائمة تتضمن اسم الطالب بالكامل كما يلي:

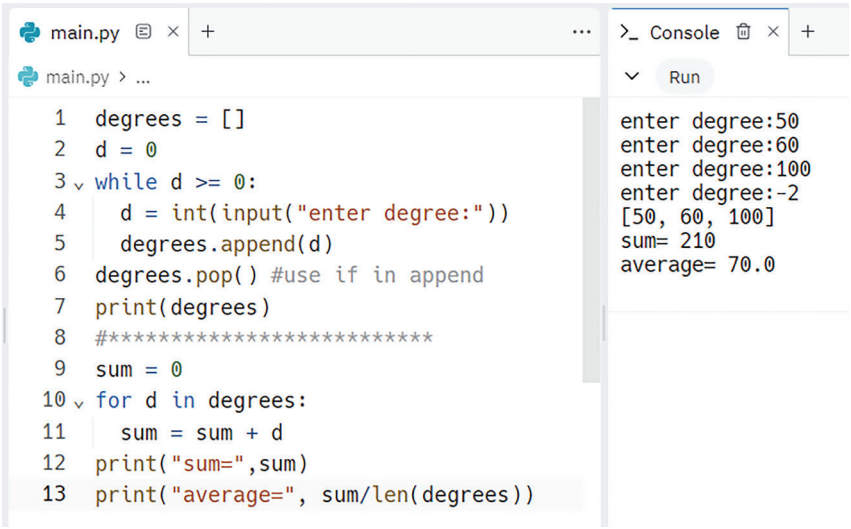


اطبع اسم الطالب بالكامل (Abdulrahman Ahmed Ali)

مثال (1-25)

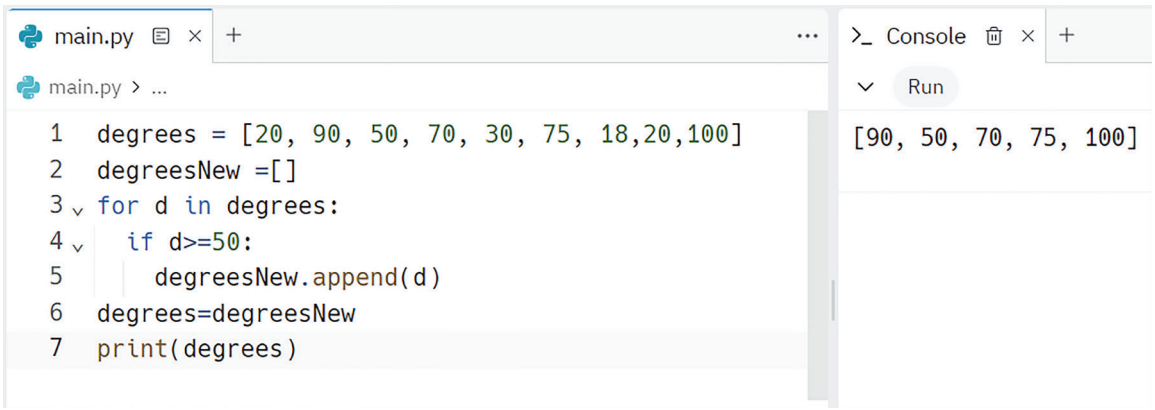
احسب مجموع الدرجات، والمتوسط الحسابي بعد إدخال مجموعة من درجات الطلاب.

لاحظ: لمعرفة انتهاء عملية إدخال الدرجات من خلال إدخال رقم سالب.

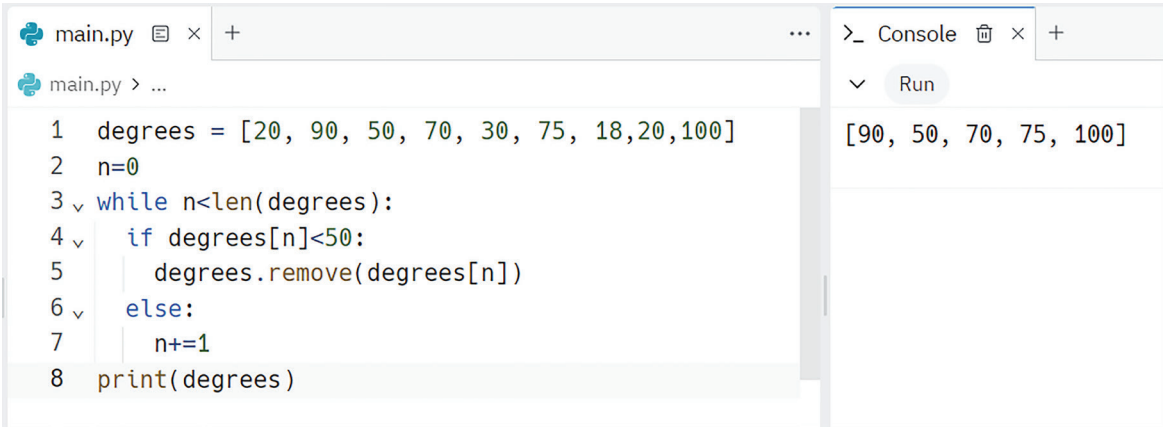


مثال (1-26)

لديك سلسلة تتضمن مجموعة من درجات الطلاب، المطلوب حذف جميع الدرجات الأقل من 50 في السلسلة.



حل آخر:



2- الصفوف Tuples

هي أحد أنواع البيانات الأساسية في بايثون. وهي عبارة عن صفوف مرتبة من البيانات يمكن أن تحتوي على أي نوع من البيانات، بما في ذلك الأعداد الصحيحة والأرقام العشرية والسلاسل النصية والكائنات.

الصيغة Syntax	أنواع مختلفة من البيانات Different of Data Types	العناصر الفريدة Unique Elements	الفهرسة Indexing	العناصر مرتبة Ordered	قابلة للتغيير Mutable	نوع البيانات Data Type
() Parentheses	نعم	لا	نعم	نعم	لا	Tuple

لذلك أي مجموعة من البيانات والتي نحتاج إلى عدم تغيير قيمتها بالخطأ أثناء تنفيذ البرنامج يستخدم tuple مثال أيام الأسبوع: الأحد، الاثنين، ... إلخ يفضل حفظها في tuple.

ورقة عمل (1-1)

برنامج رموز العناصر الكيميائية

مشكلة البرنامج:

تحديد رموز العناصر الكيميائية المختلفة.

الخوارزمية:

- الإعلان عن قائمة بأسماء العناصر الكيميائية element_name.
- الإعلان عن قائمة برموز العناصر element_symbole بنفس ترتيب القائمة الأولى.
- الإعلان عن المتغير degree، وقيمته الابتدائية تساوي صفرًا لحساب درجة الطالب degree = 0.
- الإعلان عن المتغير es لاستقبال رموز العناصر من المستخدم، وتكون قيمته الابتدائية ' '.
- إنشاء حلقة تكرارية بعدد عناصر القائمة.

أ. يُظهر عنصر من القائمة element_name ويطلب من المستخدم إدخال رمزه.

ب. يختبر تطابق الرمز المُدخل مع نظيره في القائمة element_symbol ثم:

- يُظهر عبارة «Correct» إذا كانت الإجابة صحيحة، ويضيف 1 إلى قيمة المتغير degree.
- يُظهر عبارة «Error» إذا كانت الإجابة خاطئة، بدون تغيير في قيمة المتغير degree.

- يظهر الدرجة الكلية.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- افتح الملف chemistry.py وقارنه بالخوارزمية السابقة.
- صحح الأخطاء، ثم استكمل التعليمات البرمجية ليعمل البرنامج بشكل صحيح.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

main.py > ... Format

```
1  element_name=['Sodium','Potassium','Copper','Iron','Mercury','Lead']
2  element_symbol=
3  x=0
4  es=""
5  degree=0
6  while x<=:
7
8
9
10 print('degree', degree , "/6")
```

اسم العنصر الكيميائي	Sodium	Potassium	Copper	Iron	Mercury	Lead
الرمز	Na	K	Cu	Fe	Hg	Pb

جدول (1-1) قائمة ببعض العناصر الكيميائية ورموزها.

ورقة عمل (1-2)

برنامج المتوسط الحسابي

مشكلة البرنامج:

إنشاء قائمة فارغة، وإدخال عناصرها العددية من المستخدم، ثم حساب المتوسط الحسابي للعناصر.

الخوارزمية:

- الإعلان عن قائمة فارغة.
- الإعلان عن متغير فارغ sum لجمع عناصر القائمة.
- إنشاء حلقة تكرارية لا نهائية لتنفيذ التالي :
  - استقبال من المستخدم عدد.
  - إذا كان عملية الإدخال تساوي (تم).. يخرج من الحلقة التكرارية.
  - إذا لم يتحقق الشرط السابق: يحول المدخل إلى رقم عشري، ويضيفه للقائمة.
  - جمع العنصر المدخل في المتغير sum.
- حساب المتوسط الحسابي الذي يساوي (مجموع عناصر القائمة مقسومًا على طول القائمة).
- طباعة عبارة (الأعداد التي قمت بإدخالها).
- طباعة القائمة.
- طباعة عبارة (المتوسط الحسابي) متبوعة بالمتوسط الحسابي الذي تم حسابه.

المطلوب:

خريطة التدفق

- ارسم خريطة التدفق المناسبة.

البرنامج

- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

ورقة عمل (1-3)

برنامج مدير المهام

مشكلة البرنامج:

توظيف بعض الدوال المتعلقة بالقوائم، (الطباعة، والإضافة).

الخوارزمية:

- يظهر للمستخدم ثلاث خيارات:

1- إدخال مهمة جديدة .

2- عرض المهام.

3- خروج من البرنامج.

- يستقبل من المستخدم الخيار المناسب:

1- إذا اختار المستخدم **1**: (يستقبل من المستخدم المهمة الجديدة، ويضيفها إلى القائمة)، ويظهر رسالة «تمت إضافة المهمة بنجاح»، ثم يستمر التكرار.

2- إذا اختار المستخدم **2**: (يطبع محتوى القائمة على الشاشة).، يظهر عبارة «مهامك هي»، ثم يعرض القائمة، ثم يستمر التكرار.

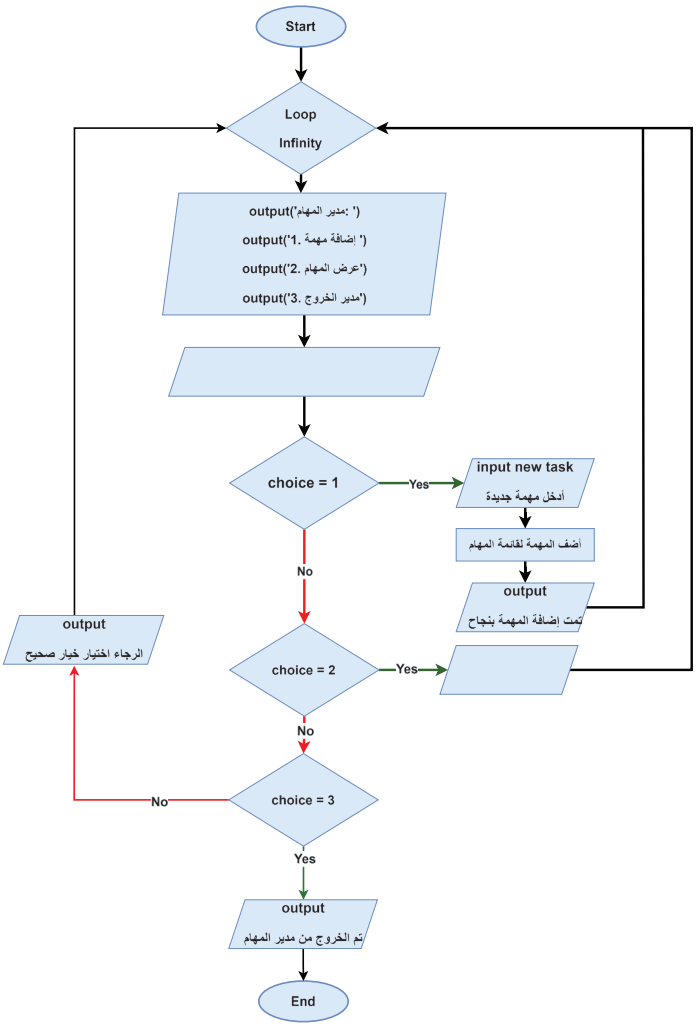
3- إذا اختار المستخدم **3**: (يخرج من البرنامج).

4- إذا أدخل المستخدم عددًا مختلفًا عن الخيارات السابقة، يُظهر رسالة تحذير بخطأ في الإدخال.

المطلوب:

خريطة التدفق

- ادرس خريطة التدفق التالية، ثم أكمل الناقص.



شكل (1-1) خريطة تدفق برنامج مدير المهام

البرنامج

- اكتب التعليمات البرمجية اللازمة.
- اختبر البرنامج، وتأكد من خلوه من الأخطاء.

برمجة Python

المنتجات الرقمية

نواتج التعلم

- الاستخدام الأمثل للمهارات البرمجية في لغة Python.
- تنمية مهارات الابتكاروالإبداع، من خلال تحليل المشكلات.
- توظيف المهارات البرمجية في تطوير برامج حياتية.
- تطوير مهارات العمل الجماعي، وتبادل الخبرات.
- تنمية مهارات العرض والمناقشة وتقبل الرأي الآخر.
- الحصول على التغذية الراجعة، وتحسين المشروع.

يمثل رمز الاستجابة السريعة QR رابط
لملفات أوراق العمل، ومصادر التعلم.

المنتجات الرقمية

من خلال دراستك لوحدة البرمجة بلغة البايثون، والاطلاع على التطبيقات (المنتجات الرقمية) المساعدة المتاحة على الرابط الإلكتروني المتمثل في رمز الاستجابة السريع QR السابق، والمتوافر على أجهزة الحاسوب بالمختبر المدرسي، صمم المشروع الخاص بك، والذي يتضمن كلًا من:

- تحديد مشكلة المشروع.
- إعداد الخوارزمية.
- تصميم خريطة التدفق.
- كتابة الرموز البرمجية.
- اختبار البرنامج، تصحيح البرنامج إن وجد.
- عرض المشروع على زملائك، ومعلمك.
- تحسين البرنامج وفق التغذية الراجعة.
- بناء البرنامج.
- بناء البرنامج، وتوثيق البرنامج.

المراجع

- مؤسسة بايثون للبرمجيات. (2024). دليل بايثون الرسمي: إصدار 7. https://www.python.org/doc.
- إطار عمل الأمن السيبراني الوطني الأمريكي. (2024). https://www.nist.gov/cyberframework.