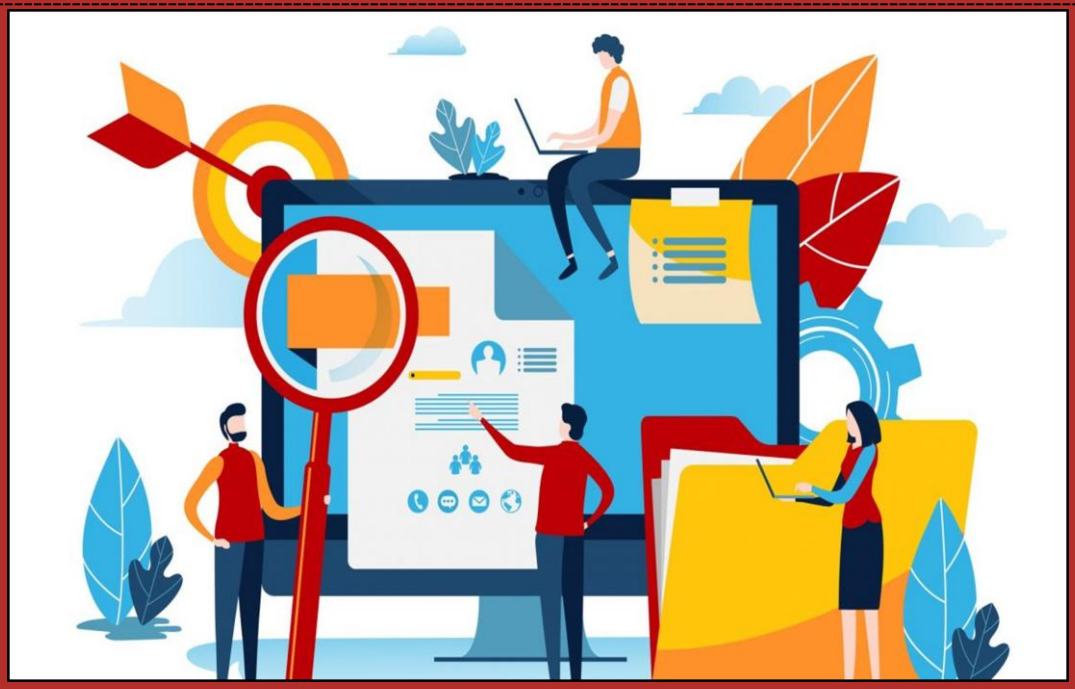

الوحدة التدريبية السابعة
المساهمة في اختبار وتصحيح التطبيق وتوثيقة

رمز الوحدة
ADS-U7-CAVT

المتطلبات السابقة:

لا يوجد

نتاجات التعلم:

عند الانتهاء من دراسة هذه الوحدة واكتساب مهاراتها الادائية والاتجاهات السلوكية الصحيحة والعلوم المهنية المرافقة من خلال التفاعل مع أنشطتها وخبراتها المختلفة يصبح المتدرب قادراً على أداء نتاجات التعلم الآتية:

1. يقدم الاقتراحات في تحضير خطة الاختبار
2. المساهمة في إجراءات الاختبار المناسبة لنصوص الاختبار التي تم اعدادها
3. المساهمة في تصحيح برنامج التطبيق وتوثيق التصحيحات

الأنشطة التعليمية والتدريبية	مصادر التعلم
قراءة المعلومات النظرية	الوحدة التدريبية
البحث في المواقع الالكترونية التعليمية روابط التعليم الالكتروني	الشبكة العنكبوتية منصة الكلية التعليمية
تنفيذ التمارين العملية	المشغل / المختبر
التدريب الميداني	سوق العمل

روابط التعلم الإلكتروني

سيتم تزويد المتدربين بروابط التعلم الإلكتروني من خلال المدرب

- كيف يمكن لتصحيح الأخطاء الفعال والتوثيق الشامل تحسين جودة البرامج؟

- **تصحيح الأخطاء وأهميتها في تطوير البرمجيات:**
- **مقدمة في تصحيح الأخطاء:**

تصحيح الأخطاء (Debugging) هو عملية تحديد الأخطاء أو الأخطاء في البرامج وتحليلها وإزالتها. إنه جانب أساسي في تطوير البرمجيات، مما يضمن أن التطبيق يعمل بشكل صحيح ويلبي المتطلبات المحددة.

- **أهمية التصحيح:**

1. ضمان جودة البرمجيات:

-يساعد تصحيح الأخطاء في اكتشاف وإصلاح الأخطاء التي يمكن أن تتسبب في حدوث خلل في البرنامج. ومن خلال معالجة هذه المشكلات، يضمن المطورون أن البرنامج يعمل على النحو المنشود، مما يؤدي إلى زيادة الجودة والموثوقية.

- كما أنه يساعد في الحفاظ على أداء البرنامج، مما يضمن تشغيله بكفاءة دون حدوث أعطال أو تباطؤ غير متوقع.

2. يعزز تجربة المستخدم:

-يمكن أن تؤدي أخطاء البرامج إلى تجارب مستخدم محبطة، مما يتسبب في عمل التطبيقات بشكل غير متوقع أو فشلها. ومن خلال تصحيح الأخطاء بشكل فعال، يمكن للمطورين توفير تجربة أكثر سلاسة ومتعة للمستخدمين.

-يمكن أن يؤدي إصلاح الأخطاء على الفور إلى منع ردود الفعل السلبية وتحسين رضا المستخدمين والاحتفاظ بهم.

3. يقلل من تكاليف الصيانة:

- يعد تحديد الأخطاء وإصلاحها في وقت مبكر من عملية التطوير أكثر فعالية من حيث التكلفة من معالجتها لاحقًا. يساعد تصحيح الأخطاء في اكتشاف المشكلات قبل أن تتفاقم إلى مشكلات أكثر أهمية، مما يقلل الحاجة إلى إعادة العمل على نطاق واسع.
- من السهل تحديث وتوسيع التعليمات البرمجية التي يتم صيانتها جيدًا والتي تحتوي على عدد أقل من الأخطاء، مما يقلل من تكاليف الصيانة على المدى الطويل.

4. تحسين فهم التعليمات البرمجية:

- يتضمن تصحيح الأخطاء فحص التعليمات البرمجية عن كثب، مما يساعد المطورين على فهم قاعدة التعليمات البرمجية بشكل أفضل. يمكن أن يؤدي هذا الفهم الأعمق إلى ممارسات ترميز أفضل وتصميم برمجيات أكثر قوة.
- إنه يشجع المطورين على كتابة تعليمات برمجية أكثر وضوحًا وأكثر قابلية للصيانة، حيث يصبحون أكثر وعيًا بالمزالق المحتملة والأخطاء الشائعة.

5. يسهل تعاون الفريق:

- غالبًا ما يتطلب تصحيح الأخطاء التعاون بين أعضاء الفريق لتحديد المشكلات وحلها. يمكن أن يؤدي هذا الجهد التعاوني إلى تحسين التواصل والعمل الجماعي داخل فريق التطوير.
- يمكن أن تساعد مشاركة المعرفة والتقنيات أثناء جلسات تصحيح الأخطاء أعضاء الفريق على التعلم من بعضهم بعضاً وتعزيز مهاراتهم في حل المشكلات.

- **أدوات وتقنيات التصحيح الشائعة**

تعد أدوات وتقنيات تصحيح الأخطاء ضرورية لتحديد الأخطاء في البرامج وإصلاحها بكفاءة. تساعد هذه الأدوات المطورين على تحليل التعليمات البرمجية وتعقب الأخطاء وفهم سلوك التطبيق.

- **أدوات التصحيح الشائعة:**

1. بيئات التطوير المتكاملة ((Integrated Development Environments (IDEs):

- **Visual Studio Code**: محرر أكواد برمجية شائع يتمتع بقدرات تصحيح أخطاء مضمنة لمختلف لغات البرمجة.
- **Eclipse**: بيئة تطوير متكاملة (IDE) شائعة الاستخدام لتطوير Java، وتوفر أدوات تصحيح أخطاء قوية.
- **IntelliJ IDEA**: بيئة تطوير متكاملة (IDE) معروفة بميزات تصحيح الأخطاء القوية، خاصة لتطوير Java و Kotlin .

2. المصححات (Debuggers):

- **GDB (GNU Debugger)**: مصحح أخطاء قوي للبرامج المكتوبة بلغات C وC++ ولغات أخرى. فهو يسمح للمطورين برؤية ما يحدث داخل برنامجهم أثناء تشغيله أو ما كان يفعله لحظة تعطله.
- **LLDB**: مصحح أخطاء يعد جزءًا من مشروع LLVM، وهو مناسب لتصحيح أخطاء البرامج المكتوبة بلغات C وC++ وObjective-C.

3. أدوات التسجيل (Logging Tools):

- **Log4j** : أداة مساعدة للتسجيل تعتمد على Java وتساعد في تتبع تدفق التطبيق وتحديد المشكلات.
- **Winston** : مكتبة تسجيل متعددة الاستخدامات لتطبيقات Node.js.

4. أدوات التنميط (Profiling Tools):

- **VisualVM** : أداة توفر معلومات تفصيلية حول تطبيقات Java، بما في ذلك ملفات التعريف وتحليل تفريغ الكومة ومراقبة جمع البيانات المهملة.
- **JProfiler** : أداة ملفات تعريف Java التي تساعد في تحديد اختناقات الأداء وتسرب الذاكرة.

- **تقنيات التصحيح الشائعة:**

1. إدراج عبارات الطباعة (Print Debugging):

- **الوصف** : إدراج عبارات الطباعة في التعليمات البرمجية لعرض القيم المتغيرة وتدفق التنفيذ وحالة البرنامج.
- **الاستخدام** : مفيد للتحقق بسرعة من حالة البرنامج دون استخدام مصحح الأخطاء. يشيع استخدامها أثناء تحديد الأخطاء الأولية.

مثال:

```
python
    def calculate_sum(a, b):
        print(f"Calculating sum of {a} and {b}")
        result = a + b
        print(f"Result: {result}")
        return result
```

2. نقاط التوقف (Breakpoints):

- **الوصف** : ميزة تصحيح الأخطاء التي تسمح للمطورين بإيقاف تنفيذ البرنامج مؤقتًا عند نقاط محددة لفحص حالة التطبيق.
- **الاستخدام** : قم بتعيين نقاط التوقف في IDE أو مصحح الأخطاء لإيقاف التنفيذ وفحص المتغيرات والذاكرة ومكدسات الاستدعاءات.

مثال:

- في Visual Studio Code، انقر فوق الحضيض بجوار رقم السطر لتعيين نقطة توقف. قم بتشغيل مصحح الأخطاء لإيقاف التنفيذ مؤقتًا عند نقطة التوقف.

3. تنفيذ الخطوة (Step Execution):

-**الوصف** : تنفيذ التعليمات البرمجية خطوة بخطوة لمراقبة سلوك البرنامج وتحديد مكان حدوث المشكلات.

-**الاستخدام** : استخدم أوامر "Step Over" و"Step Into" و"Step Out" في مصحح الأخطاء للتحكم في تدفق التنفيذ.

مثال:

- استخدم الأمر "Step Over" لتنفيذ السطر الحالي والانتقال إلى السطر التالي، دون الدخول في أي استدعاءات دالة.

4. التفتيش المتغير (Variable Inspection):

- **الوصف** : فحص قيم المتغيرات في نقاط مختلفة أثناء تنفيذ البرنامج لتحديد القيم غير الصحيحة أو السلوك غير المتوقع.
- **الاستخدام** : استخدم ميزة الفحص المتغير الخاصة بمصحح الأخطاء للتحقق من الحالات المتغيرة وتعديل قيمها أثناء وقت التشغيل.

مثال:

- في Eclipse، استخدم عرض "المتغيرات" لفحص قيم المتغيرات وتعديلها أثناء تصحيح الأخطاء.

5. مر اقبة التعبيرات (Watch Expressions):

-**الوصف** : مراقبة تعبيرات أو متغيرات محددة لمعرفة كيفية تغير قيمها أثناء تنفيذ البرنامج.

-**الاستخدام** : أضف تعبيرات المراقبة في مصحح الأخطاء لتتبع القيم المتغيرة بشكل مستمر واكتشاف الحالات الشاذة.

مثال:

- في IntelliJ IDEA، انقر بزر الماوس الأيمن على متغير وحدد "إضافة إلى الساعات" لمراقبة قيمته.

6. معالجة الاستثناءات (Exception Handling):

-**الوصف** : استخدام كتل محاولة الالتقاط للتعامل مع الاستثناءات وتقديم رسائل خطأ أو خيارات استرداد ذات معنى.

-**الاستخدام** : تنفيذ معالجة الاستثناءات لاكتشاف الأخطاء وتسجيلها، مما يسهل تشخيص المشكلات وإصلاحها.

مثال:

```
java
    try {
        int result = 10 / 0;
    } catch (ArithmeticException e) {
        System.out.println("Error: Division by zero is not allowed.");
    }
```

خاتمة:

يعد تصحيح الأخطاء جزءًا مهمًا من عملية تطوير البرامج، مما يضمن عمل التطبيقات بشكل صحيح وتلبية متطلبات المستخدم. من خلال فهم أهمية تصحيح الأخطاء واستخدام الأدوات والتقنيات الشائعة، يمكن للمطورين تحديد المشكلات وحلها بكفاءة، مما يؤدي إلى برامج عالية الجودة وتحسين تجارب المستخدم.

- **التمارين العملية**

اسم التمرين: **تطوير خطة اختبار شاملة لتطبيق برمجي** رقم التمرين: 1/9 الزمن المخصص لتنفيذ التمرين:
اهداف التمرين العملي: يهدف التمرين الى 1. فهم أهمية اختبار البرمجيات: • ضمان الجودة: التأكد من أن التطبيق يلبي المتطلبات المحددة ويعمل بشكل صحيح. • اكتشاف الأخطاء: تحديد الأخطاء والمشاكل في التطبيق قبل إطلاقه • تحسين تجربة المستخدم: ضمان أن التطبيق سهل الاستخدام ويعمل بسلاسة. 2. تحديد أنواع الاختبارات: 3. تطوير خطة اختبار شاملة 4. إنشاء حالات الاختبار (Test Cases) 5. تنفيذ الاختبارات 6. تحليل النتائج وإعداد التقارير 7. تحسين عملية الاختبار 8. التوثيق 9. التعلم الذاتي والتوسع
تعليمات المدرب: **يقدم المدرب توضيح للخطوات حسب تعليمات السلامة المتبعة في المملكة الاردنية الهاشمية ويراعي ان يوضح كل من التعليمات التالية والتي تتفق مع اجراءات السلامة حسب القو انين المرعية.** • **الخروج من موقع العمل عند سماع جرس الانذار.** • **اسلك الطرف الايسر من الممرات الامنة و اترك الطرف الايمن خاليا** • **عند الخروج من الغرف تفقد اتجاه الخطر قبل الخروج مثلا تفقد يد الباب عند الحريق للتأكد من عدم وجود حرارة مرتفعة.** • **اغلاق الباب عند الخروج من الاماكن المغلقة لمنع انتشار الحريق** • **الصراخ عند الخروج او استخدام مكبر الصوت منبها الاخرين بكلمة اخلاء مع التكرير عند كل نقطة** • **الاتصال بالدفاع المدني اذا تطلبت الحاجة يوضع رقم الدفاع المدني امام المتدربين بطريقة ظاهرة , رقم الطوارئ الموحد 911 .**

- **التوجه الى مو اقع التجمع المحددة.**
- **تفقد فريق عملك وحدد الغائبين.**
- **ابلاغ فريق الانقاذ بأعداد الحضورواسماء الغائبين.**

التسهيلات التدريبية اللازمة لتنفيذ التمرين العملي :

الاجهزة والآلات	العدد والادوات	المواد الاولية
✓ أدوات الاتصال والمراسلة • أدوات ادارة المشروع • أدوات التحرير والتوثيق مثلMicrosoft Office • منصات التواصل الاجتماعي أدوات لتحليل أنماط التعاون ومقاييس الاتصال وتقدم المشروع مثل , Google Analytics Microsoft Power point	أجهزة الحاسوب، الشاشات، انترنت عالي السرعة، سماعات الرأس، كاميرات الويب، الواح تفاعلية، طابعات، ماسحات ضوئية	• قرطاسية (ورق للكتابة ، اقلام حبر، اقلام تخطيط • أدوات العصف الذهني مثل السبورة الورقية Flip Chart

خطوات العمل لتنفيذ التمرين

خطوات العمل ومعايير ادائها (الرقمية & الوصفية)	الشكل / الصورة
حدد الغرض الرئيسي من التطبيق والفئة المستهدفة	
ناقش مع المطورين ومحللي الأعمال لفهم التوقعات وأي تفاصيل فنية	
قم بتقسيم التطبيق إلى وحدات أو مكونات رئيسية لتحديد ما سيتم اختباره	
حدد أنواع الاختبارات المطلوبة (وظيفية، أداء، أمان، واجهة مستخدم، توافق)	
رتب المهام حسب أهميتها وتعقيدها	

حدد ما إذا كان الاختبار سيكون يدويًا أو آليًا أو مزيجًا من الاثنين	
حدد كيفية اختبار الوحدات الفردية واختبار تكاملها مع بعضها	
اختر الأدوات المناسبة للاختبار	
حدد أعضاء الفريق المسؤولين عن تنفيذ الاختبارات	
جهز بيئة الاختبار	
وثق نتائج كل اختبار (نجاح/فشل) وأي ملاحظات	
جمع تعليقات الفريق لتحسين خطط الاختبار المستقبلية	
تقييم جودة المنتج: وهي مواصفات جودة ودقة ناتج الأداء المطلوب سواء كان منتج أو خدمة	
وضوح الأهداف والتخطيط	
تفاعل المشاركين والمشاركة الفعالة	
جودة المحتوى والأدوات المستخدمة	
تأثير التمرين على المشاركين	
التقييم والتغذية الراجعة	
تقييم الاتجاهات والسلامة المهنية: وهي الاتجاهات المتعلقة بالمحافظة على الأجهزة والأدوات وترتيبها وتنظيفها وتخزينها والسلامة المهنية المتعلقة بالأشخاص والآلات والمعدات والمواد أثناء تأدية التمرين العملي	
الوعي والالتزام بقواعد السلامة	
السلوكيات والممارسات اليومية	
الثقافة العامة للسلامة المهنية	
تقييم بيئة العمل	
الالتزام باللوائح والقوانين	
التدريب والتطوير	

تقييم التمرين العملي رقم (1 / 9) **ترفق بعد كل بطاقة تمرين عملي**

قيم ادائك بعد انتهائك من تنفيذ التمرين من خلال:

- ضع اشارة " ✓ " بجانب الخطوات التي اتقنتها.
- ضع إشارة " × " بجانب الخطوات التي لم تتقنها مع ذكر الاسباب وكيف يمكنك تحسينها بالمناقشة مع مدربك
- ضع إشارة " × " بجانب الخطوات غير القابلة للتطبيق مع ذكر الاسباب المؤدية لعدم قابلية تطبيق وتنفيذ الخطوة وكيف يمكنك اعادة تطبيقها باتقان وتحسينها بالمناقشة مع مدربك.

الرقم	اسم الخطوة **(عليك اتقان جميع الخطوات أدناه حتى تظهر درجة الإتقان للأداء المطلوب في التمرين)**	اتقنها	لا اتقنها	غير قابلة للتطبيق	ذكر اسباب عدم الاتقان ولماذا هي غير قابلة للتطبيق
1	هل تم تحديد هدف التمرين				
2	هل استطاع شرح أهمية تطوير خطة اختبار شاملة لتطبيق برمجي				
3	هل قام بتحليل استخدام الوقت الحالي				
4	هل قام بتقديم أدوات وتقنيات تطوير خطة اختبار شاملة لتطبيق برمجي				
5	هل تم توضيح الأهداف والتخطيط				
6	هل كان جودة المحتوى والأدوات المستخدمة جيدة				
7	هل هنالك التزام بقواعد السلامة				
8	السلوكيات والممارسات اليومية				
9	الثقافة العامة للسلامة المهنية				
10	هل بيئة العمل جيدة				
11	هل تم الالتزام باللوائح والقوانين				
أنجزت اداء التمرين العملي المطلوب		ضمن الوقت المحدد	زيادة عن الوقت المحدد	غير قابل للإنجاز	ذكر اسباب: ✓ زيادة عن الوقت المحدد ✓ ولماذا التمرين غير قابل للإنجاز