**Отчет**

**Выполнила: Чулюкина Татьяна ИСП-21**

**Экзамен по МДК.02.02. Инструментальные средства разработки программного**

**Обеспечения**

**Междисциплинарный курс МДК.02.02. Инструментальные средства разработки**

**программного обеспечения**

**Специальность 09.02.07 Информационные системы и программирование**

**БИЛЕТ НОМЕР 25**

1. **Принципы отладки ПО.**
   * Большинство ошибок обнаруживается вообще без запуска программы - просто внимательным просматриванием текста.
   * Если отладка зашла в тупик и обнаружить ошибку не удается, лучше отложить программу. Когда глаз "замылен", эффективность работы упорно стремится к нулю.
   * Чрезвычайно удобные вспомогательные средства - это отладочные механизмы среды разработки: трассировка, промежуточный контроль значений. Можно использовать даже дамп памяти, но такие радикальные действия нужны крайне редко.
   * Экспериментирования типа "а что будет, если изменить плюс на минус" - нужно избегать всеми силами. Обычно это не дает результатов, а только больше запутывает процесс отладки, да еще и добавляет новые ошибки.

* Там, где найдена одна ошибка, возможно, есть и другие.
* Вероятность, что ошибка найдена правильно, никогда не равна ста процентам.
* Наша задача - найти саму ошибку, а не ее симптом.

1. **Ручное и автоматизированное тестирование. Плюсы и минусы.**

***Ручное тестирование*** – это прямое взаимодействие QA-инженера и приложения. В его процессе можно получить обратную связь о продукте, что невозможно, если использовать автоматизированное тестирование.

Кроме того, тестировщик сможет составить свой отзыв и рекомендации по улучшению ПО, а проведенное тестирование будет свидетельством сравнения ожидаемого и реального результатов.

***Плюсы ручного тестирования:***

* Отчет тестировщика – это первый отзыв потенциального клиента, который позволит понять, насколько продукт удобен для конечного пользователя.
* Обратная связь по UI. Протестировать общий дизайн приложения и выявить его недостатки представляется возможным только при ручном тестировании.
* Стоимость. Когда речь идет о небольшом проекте, внедрять ручное тестирование всегда менее затратно, чем автоматизацию.
* Гибкость. Тестирование несущественных изменений происходит сразу, без затрат на написание кода. Это особенно важно при быстром внедрении новой функциональности, когда нужно быть уверенным в ее корректной работе.
* Исследовательское тестирование и возможность импровизации позволяет проверить потенциал приложения в нетипичных сценариях и обнаружить существенные дефекты в короткие сроки.

***Минусы ручного тестирования:***

* Человеческий фактор. Часть ошибок продукта может быть пропущена, а некоторые результаты проверки могут оказаться субъективными.
* Трудозатраты и продолжительность. Серия автоматизированных тестов позволяет протестировать программное обеспечение значительно быстрее.
* Отсутствие возможности моделирования большой нагрузки. При ручном тестировании невозможно смоделировать большое количество пользователей.

***Автоматизация тестирования*** приложений основана на написании кода. Методика автоматизированного тестирования позволяет создать ожидаемый сценарий, а затем сравнить его с реальным и указать расхождения.

Автоматизация наиболее применима в сложных приложениях с большой функциональной частью. Особой популярностью пользуется автоматизация тестирования труднодоступных мест приложения, валидационных форм, базовых операций, часто используемой функциональности.

***Плюсы автоматизированного тестирования:***

* Нагрузка на приложение. Когда используется автоматизированное тестирование, становится возможным моделирование большой нагрузки, которая приближена к реальной ситуации.
* Временной фактор. Ручное тестирование – это долгий и ресурсоемкий процесс, в то время как код для сценария пишется один раз.
* Повторяемость. Код автотестов может быть использован неоднократно, особенно при внедрении новой функциональности.

***Минусы автоматизированного тестирования:***

* Отсутствие обратной связи. Автоматизированное тестирование не способно предоставить обратную связь относительно качества продукта – оно лишь выполняет запрограммированные сценарии.
* Отсутствие тестирования глазами пользователя. Иногда в приложении остаются ошибки, которые могут быть не покрыты автотестами.
* Отсутствие возможности тестирования цвета, дизайна и эргономики. Этот пункт не является первостепенным, но может значительно повлиять на качество продукта.
* Надежность. Автоматизированные тесты могут упасть по многим причинам, например, при большой загруженности тестовой машины или при проблемах с сетью.
* Стоимость. Для небольших проектов инструменты автоматизированного тестирования могут оказаться достаточно затратными, поэтому более рационально их использовать для долгосрочных проектов.

1. **Написать программу с использованием классов. Определить минимальный элемент массива А[15] и его порядковый номер.**

{

public class A

{

public void kto()

{

int min\_ind = 0;

int[] A = new int[15];

Random r = new Random();

for (int i = 0; i < A.Length; i++)

A[i] = r.Next(0, 100);

for (int i = 0; i < A.Length; i++)

Console.Write(A[i] + " ");

Console.WriteLine();

int min;

min = A[0];

for (int i = 1; i < A.Length; i++)

{

if (min > A[i])

{

min = A[i];

min\_ind = i;

}

}

Console.WriteLine("\nМинимальное значение: " + min + "\nИндекс значения: " + min\_ind);

Console.ReadKey();

}

}

class Program

{

static void Main(string[] args)

{

A mas = new A();

mas.kto();

}

}

}