**Лабораторное занятие №1**

**Тема:** тестирование программ методами «белого ящика».

**Цели занятия:**

- освоить методы тестирования «белым ящиком»;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Разработала приложение для определения вида и площади треугольника.

CS

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Find\_Click(object sender, RoutedEventArgs e)

{

double S;

double p;

string type;

try

{

double a = Convert.ToInt32(text1.Text);

double b = Convert.ToInt32(text2.Text);

double c = Convert.ToInt32(text3.Text);

p = (a + b + c) / 2;

S = Math.Round(Math.Sqrt(p \* (p - a) \* (p - b) \* (p - c)),3);

if (a + b > c && a + c > b && b + c > a && a>0 && b>0 && c>0)

{

if (a == b && b == c)

{

type = "равносторонний";

}

else if (a == b || b == c || a == c)

{

type = "равнобедренный";

}

else

{

type = "разносторонний";

}

typeInput.Text = type;

SInput.Text = Convert.ToString(S);

}

else {

MessageBox.Show("Треугольника с такими сторонами не существует", "Ошибка");

}

}

catch {

MessageBox.Show("Введены неверные стороны", "Ошибка");

}

}

}

XML

<Grid>

<TextBlock HorizontalAlignment="Left" Margin="90,100,0,0" TextWrapping="Wrap" Text="a: " VerticalAlignment="Top" FontSize="20"/>

<TextBlock HorizontalAlignment="Left" Margin="343,100,0,0" TextWrapping="Wrap" Text="b: " VerticalAlignment="Top" FontSize="20"/>

<TextBlock HorizontalAlignment="Left" Margin="555,99,0,0" TextWrapping="Wrap" Text="c: " VerticalAlignment="Top" FontSize="20"/>

<TextBlock HorizontalAlignment="Center" Margin="50,27,0,0" TextWrapping="Wrap" Text="Введите длины сторон треугольника для определения его вида и площади" VerticalAlignment="Top" FontSize="20"/>

<TextBox x:Name="text1" HorizontalAlignment="Left" Margin="115,99,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="120" FontSize="20"/>

<TextBox x:Name="text2" HorizontalAlignment="Left" Margin="370,99,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="120" FontSize="20"/>

<TextBox x:Name="text3" HorizontalAlignment="Left" Margin="579,99,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="120" FontSize="20"/>

<Button x:Name="Find" FontSize="20" Content="Найти" Height="40" Width="150" Cursor="Hand" Background="#FFAAD0AE" VerticalAlignment="Top" HorizontalAlignment="Center" Click="Find\_Click" Margin="0,184,0,0" />

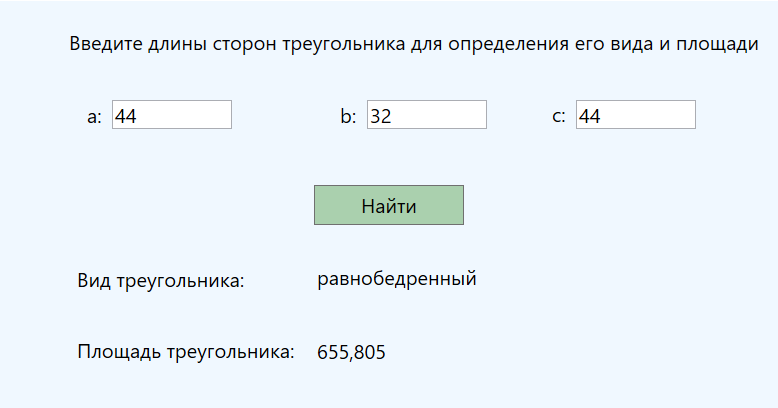
<TextBlock HorizontalAlignment="Left" Margin="80,264,0,138" TextWrapping="Wrap" Text="Вид треугольника: " FontSize="20"/>

<TextBlock x:Name="typeInput" HorizontalAlignment="Left" Margin="320,262,0,184" TextWrapping="Wrap" Text=" " FontSize="20"/>

<TextBlock HorizontalAlignment="Left" Margin="80,335,0,69" TextWrapping="Wrap" Text="Площадь треугольника: " FontSize="20"/>

<TextBlock x:Name="SInput" HorizontalAlignment="Left" Margin="320,336,0,109" TextWrapping="Wrap" Text=" " FontSize="20"/>

</Grid>



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | C | Ожидаемый результат | Итоговый результат |
| 0 | 0 | 0 | Треугольника не существует | Треугольника не существует |
| 2 | 4 | 3 | Вид: разносторонний, площадь: 2,905 | Вид: разносторонний, площадь: 2,905 |
| -2 | 4 | 3 | Треугольника не существует | Треугольника не существует |
| 10 | 10 | 10 | Вид: равносторонний, площадь: 43,301 | Вид: равносторонний, площадь: 43,301 |
| 10 | ап | 10 | Введены неверные стороны | Введены неверные стороны |
| 89 | 34 | 190 | Треугольника не существует | Треугольника не существует |
| 4- | 34 | 19 | Введены неверные стороны | Введены неверные стороны |

Задание 2. Разработала приложение для определения вида и площади треугольника.

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Find\_Click(object sender, RoutedEventArgs e)

{

double S;

double p;

string type;

try

{

double a = Convert.ToInt32(text1.Text);

double b = Convert.ToInt32(text2.Text);

double c = Convert.ToInt32(text3.Text);

double maxst;

double vt;

double tr;

if (a > b && a > c)

{

maxst = a;

vt = b;

tr = c;

}

else if (b > a && b > c)

{

maxst = b;

vt = a;

tr = c;

}

else {

maxst = c;

vt = a;

tr = b;

}

p = (a + b + c) / 2;

S = Math.Round(Math.Sqrt(p \* (p - a) \* (p - b) \* (p - c)),3);

if (a + b > c && a + c > b && b + c > a && a>0 && b>0 && c>0)

{

if ((Math.Pow(maxst, 2) == (Math.Pow(vt, 2) + Math.Pow(tr, 2))))

{

type = "прямоугольный";

}

else if ((Math.Pow(maxst, 2) > (Math.Pow(vt, 2) + Math.Pow(tr, 2))))

{

type = "тупоугольный";

}

else

{

type = "остроугольный";

}

typeInput.Text = type;

SInput.Text = Convert.ToString(S);

}

else {

MessageBox.Show("Треугольника с такими сторонами не существует", "Ошибка");

}

}

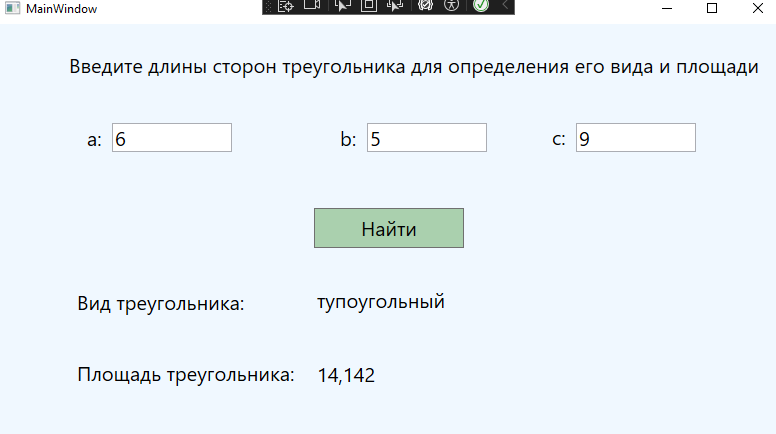
catch {

MessageBox.Show("Введены неверные стороны", "Ошибка");

}

}

}



1 тест. При вводе текстовых данных выводится сообщение выводится сообщение о том, что введены неверные стороны.

2 тест. При вводе стороны со значением 0 выводится сообщение, что треугольника с такими сторонами не существует.

3 тест. При вводе стороны с отрицательным значением выводится сообщение, что треугольника с такими сторонами не существует.

Задание 3. Разработала консольное приложение по варианту.

7) F(x)={x2 ,  если 0≤х≤3; 4, в противном случае.

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

double x, F;

try

{

x = Convert.ToDouble(textBox1.Text);

if (x >= 0 && x <= 3)

{

F = Math.Pow(x, 2);

}

else {

F = 4.0;

}

label4.Text=Convert.ToString(F);

}

catch

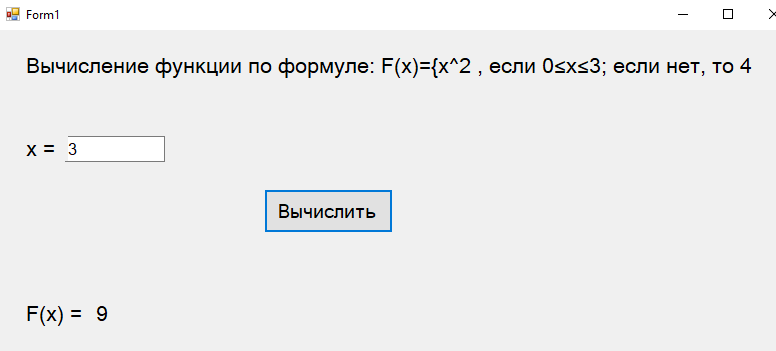
{

MessageBox.Show("Введите числа!!!", "ERROR");

}

}

}



1. Если пользователь введёт букву или другие символы и нажмёт кнопку «Найти», то будет отображено сообщение об ошибке.

2. Если пользователь ничего не введёт и нажмёт кнопку «Найти», то будет отображено сообщение об ошибке.

Задание 4. Рекомендации по исправлению ошибок: предусматривать всевозможные случаи пользовательского ввода, а также пустое значение и устанавливать конструкции проверки условия и обработку исключений.

**Лабораторное занятие №2**

**Тема:** тестирование программ методами «чёрного ящика».

**Цели занятия:**

- освоить методы тестирования «чёрным ящиком»;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Написала калькулятор, выполняющий основные арифметические операции.

public partial class Form1 : Form

{

public static double sum(double a, double b)

{

return a + b;

}

public static double razn(double a, double b)

{

return a - b;

}

public static double umnog(double a, double b)

{

return a \* b;

}

public static double delet(double a, double b)

{

return a / b;

}

public static double stepi(double a, double b)

{

return Math.Pow(a, b);

}

public static double korenn(double a, double b)

{

return Math.Pow(a, 1 / b);

}

public Form1()

{

InitializeComponent();

}

private void add\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(sum(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

{

MessageBox.Show("Введены неверные данные", "Ошибка");

}

}

private void subtr\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(razn(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

{

MessageBox.Show("Введены неверные данные", "Ошибка");

}

}

private void umn\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(umnog(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

{

MessageBox.Show("Введены неверные данные", "Ошибка");

}

}

private void del\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(delet(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

{

MessageBox.Show("Введены неверные данные", "Ошибка");

}

}

private void step\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(stepi(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

{

MessageBox.Show("Введены неверные данные","Ошибка");

}

}

private void koren\_Click(object sender, EventArgs e)

{

try

{

label1.Text = Convert.ToString(korenn(Convert.ToDouble(textBox1.Text), Convert.ToDouble(textBox2.Text)));

}

catch

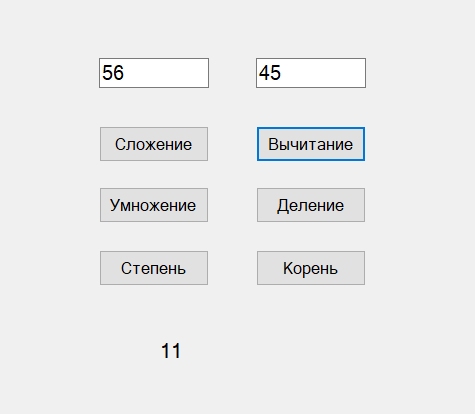
{

MessageBox.Show("Введены неверные данные", "Ошибка");

}

}

}



Задание 2. Провела тестирование и написала отчет в виде таблицы (не менее 15 сценариев).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название теста | Описание  сценария | Входные  данные | Выходные  данные | Удачное/неудачное тестирование |
| Функция суммы | Сложение двух положительных чисел; Проверка результата | Первая переменная =  3. Вторая переменная =  8 | Результат = 11 | Удачное |
| Функция вычитания | Вычитание двух отрицательных чисел; Проверка результата | Первая переменная =  -908,98. Вторая переменная = -89,764 | Результат = -819,216 | Удачное |
| Функция умножения | Умножение отрицательного и положительного числа; Проверка результата | Первая переменная =  -908,98. Вторая переменная = 89,764 | Результат = -81593,68072 | Удачное |
| Функция деления | Деление двух положительных чисел; Проверка результата | Первая переменная =  5675434356,3522. Вторая переменная = 34633532 | Результат = 163,87108182764 | Удачное |
| Функция корня | Корень отрицательного и положительного чисел; Проверка результата | Первая переменная =  -908,98. Вторая переменная = -89,764 | Результат = -819,216 | Удачное |
| Функция степени | Степень двух отрицательных чисел; Проверка результата | Первая переменная =  -56,89. Вторая переменная = -9,8 | Результат = не число | Неудачное |
| Функция корня | Корень положительного числа с нулём; Проверка результата | Первая переменная =  56,89. Вторая переменная = 0 | Результат = ∞ | Неудачное |
| Функция деления | Деление положительного числа и нуля; Проверка результата | Первая переменная =  786. Вторая переменная = 0 | Результат = ∞ | Неудачное |
| Функция сложения | Сложение положительного числа и букв; Проверка результата | Первая переменная =  786. Вторая переменная = ор | Результата нет, вывод окна с ошибкой | Удачное |
| Функция корня | Корень отрицательного числа с положительным; Проверка результата | Первая переменная =  -786. Вторая переменная = 3 | Результат = не число | Неудачное |
| Функция степени | Степень нуля и отрицательного числа; Проверка результата | Первая переменная =  0. Вторая переменная = -3 | Результат = ∞ | Удачное |
| Функция степени | Степень двух отрицательных чисел; Проверка результата | Первая переменная =  -34,9. Вторая переменная = -8,3 | Результат = не число | Неудачное |
| Функция корня | Корень двух отрицательных чисел; Проверка результата | Первая переменная =  -34,9. Вторая переменная = -8,3 | Результат = не число | Неудачное |
| Функция деления | Деление нуля и отрицательного числа; Проверка результата | Первая переменная =  0. Вторая переменная = -8,3 | Результат = 0 | Удачное |
| Функция умножения | Умножение нуля с минусом и отрицательного числа; Проверка результата | Первая переменная =  0-. Вторая переменная = -8,3 | Результата нет, вывод окна с ошибкой | Удачное |

Задание 3. Рекомендации по исправлению ошибок: предусматривать всевозможные случаи пользовательского ввода, а также пустое значение и устанавливать конструкции проверки условия и обработку исключений.

**Лабораторное занятие №3**

**Тема:** планирование тестирования.

**Цели занятия:**

- освоить методы разработки проектной документации для тестирования программного обеспечения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

**Задание 1.** План тестирования разработанной программы:

**Проект:** учёт информации о сотрудниках.

**План тестирования (Тест-план)**

Содержание

[1. Введение](about:blank)

[1.1. Цель](about:blank)

[2. Требования к тестированию](about:blank)

[3. Стратегия тестирования](about:blank)

[3.1. Типы тестирования](about:blank)

[3.2. Инструментарий](about:blank)

[4. Критерии тестирования](about:blank)

[5. Итоговые отчёты](about:blank)

**1 ВВЕДЕНИЕ**

1.1. Цель

Документ «План тестирования» для проекта «Учёт информации о сотрудниках» (далее «Продукт») преследует следующие цели:

* определить стратегии тестирования Продукта, которые планируется использовать;
* определить компоненты Продукта, которые должны быть протестированы;
* спланировать процесс тестирования и техническую поддержку тестирования;
* описать структуру и рамки тестирования, достаточные для достижения целей и решения задач тестирования в проекте;
* определить перечень инструментов, которые будут использоваться в процессе тестирования;
* определить график работ, этапы и основные вехи;
* определить обязанности, роли и ресурсы в процессе тестирования;
* определить критерии начала и окончания процесса тестирования.

**2. ТРЕБОВАНИЯ К ТЕСТИРОВАНИЮ**

 Контролю качества должны быть подвергнуты как программно-аппаратный комплекс в целом, так и его отдельные части. В ходе тестирования будут проверяться следующие комплексные показатели качества:

**Функциональные возможности:**

*Пригодность*. Свойства и характеристики программного обеспечения, относящийся к наличию и соответствию набора функций конкретным задачам. Примерами соответствия является состав функций, ориентированных на задачу, из входящих в него подфункций и объемы таблиц.

*Правильность*. Свойства и характеристики программного обеспечения, относящиеся к обеспечению правильности или соответствия результатов. Например, она включает необходимую степень точности вычисленных значений.

*Способность к взаимодействию*. Свойства и характеристики программного обеспечения, относящиеся к способности его взаимодействовать с конкретными системами.

*Согласованность*. Свойства и характеристики программного обеспечения, которые заставляют программу придерживаться соответствующих стандартов или соглашений, или положений законов, или подобных рекомендаций.

*Защищенность*. Свойства и характеристики программного обеспечения, относящиеся к его способности предотвращать несанкционированный доступ, случайный или преднамеренный, к программам и данным (*в программе отсутствует*).

**Надёжность:**

*Стабильность*. Свойства и характеристики программного обеспечения, относящиеся к частоте отказов при ошибках в программном обеспечении.

*Устойчивость к ошибкам*. Свойства и характеристики программного обеспечения, относящиеся к его способности поддерживать определенный уровень качества функционирования в случаях программных ошибок или нарушения определенного интерфейса.

*Восстанавливаемость*. Свойства и характеристики программного обеспечения, относящиеся к его возможности восстанавливать уровень качества функционирования и восстанавливать данные, непосредственно поврежденные в случае отказа, а также к времени и усилиям, необходимым для этого (*в программе отсутствует*).

**Практичность:**

*Понятность*. Свойства и характеристики программного обеспечения, относящиеся к усилиям пользователя по пониманию общей логической концепции и ее применимости.

*Обучаемость*. Свойства и характеристики программного обеспечения, относящиеся к усилиям пользователя по обучению его применению (например, оперативному управлению, вводу, выводу).

*Простота использования*. Свойства и характеристики программного обеспечения, относящиеся к усилиям пользователя, но эксплуатации и оперативному управлению.

**Эффективность:**

*Характер изменения во времени*. Свойства и характеристики программного обеспечения, относящиеся к временам отклика и обработки и к скоростям выполнения его функций.

*Характер изменения ресурсов*. Свойства и характеристики программного обеспечения, относящиеся к объему используемых ресурсов и продолжительности такого использования при выполнении функции.

**Сопровождение:**

*Анализируемость*. Свойства и характеристики программного обеспечения, относящиеся к усилиям, необходимым для диагностики недостатков или случаев отказов, или определения составных частей для модернизации.

*Изменяемость*. Свойства и характеристики программного обеспечения, относящиеся к усилиям, необходимым для модификации, устранению отказа или для изменения условий эксплуатации.

*Устойчивость*. Свойства и характеристики программного обеспечения, относящиеся к риску от непредвиденных эффектов модификации.

*Тестируемость*. Свойства и характеристики программного обеспечения, относящиеся к усилиям, необходимым для проверки модифицированного программного обеспечения.

**Мобильность:**

*Адаптируемость*. Свойства и характеристики программного обеспечения, относящиеся к удобству его адаптации к различным конкретным условиям эксплуатации, без применения других действий или способов, кроме тех, что предназначены для этого в рассматриваемом программное обеспечении.

*Простота внедрения*. Свойства и характеристики программного обеспечения, относящиеся к усилиям, необходимым для внедрения программного обеспечения в конкретное окружение.

*Соответствие*. Свойства и характеристики программного обеспечения, которые заставляют программу подчиняться стандартам или соглашениям, относящимся к мобильности.

*Взаимозаменяемость*. Свойства и характеристики программного обеспечения, относящиеся к простоте и трудоемкости его применения вместо другого конкретного программного средства в среде этого средства. Взаимозаменяемость с конкретным программным средством не предполагает, что данное средство заменимо рассматриваемым программным средством. Взаимозаменяемость может включать атрибуты простоты внедрения и адаптируемости.

**3. СТРАТЕГИЯ ТЕСТИРОВАНИЯ**

 Основными задачами тестирования являются:

* проведение функционального тестирования каждого модуля и компонента системы для проверки его соответствия функциональным требованиям;
* проведение комплексного тестирования для обеспечения взаимодействия модулей и компонентов друг с другом согласно требованиям, к системе;
* определение и максимальное увеличение производительности системы и каждого отдельного модуля;
* проведение нагрузочного тестирования для обеспечения отказоустойчивости системы и каждого отдельного модуля;
* максимальная автоматизация процесса тестирования;
* разработка достаточного набора контрольных примеров для тестирования новых модулей и компонентов;
* своевременная разработка контрольных примеров для покрытия устраняемых ошибок;
* увеличение покрытия кода тестовыми примерами;
* тестирование удобства применения модулей, имеющих графический интерфейс.

*Типы тестирования*

Для решения указанных выше задач тестирования будут использоваться следующие виды тестирования:

*Ручное тестирование — выполнение тестировщиком прохода тестового цикла вручную, с последующей ручной фиксацией результатов по каждому тесту в отчете.*

*Автоматизированное тестирование — автоматический проход тестового цикла, с последующим автоматическим уведомлением заинтересованных лиц о результатах.*

*Дымовое тестирование* — простейший вид тестирования, основанный на определении успешности сборки системы из ветви исходного кода, находящейся в разработке.

*Модульное тестирование* — самый важный вид тестирования, основанный на проверке работоспособности функций, методов и свойств в условиях их нормального и ошибочного исполнения. Это тестирование проводится на уровне исходного кода каждого существующего класса. Что нужно тестировать на данном этапе:

* класс правильно объявлен;
* структура класса соответствует спецификации требований; o класс имеет достаточную функциональность;
* класс совместим со средствами автоматической обработки кода (анализ покрытия, качества кода и т.п.);
* некорректное функционирование и ошибочные ситуации корректно обрабатываются; o класс совместим со связанными классами в рамках используемого наследования, полиморфизма, процедур вызова и т.п.;
* время выполнения, частота выполнения, нагрузка на ресурсы соответствуют требованиям; o класс не содержит утечек памяти и других ресурсов.

*Модульное интеграционное тестирование* — после разработки тестов на отдельные классы необходимо проверить, как они будут работать вместе в рамках одного исполняемого процесса. Необходимо проверить, как соотносятся классы, разработанные по-разному разными разработчиками.

*Системное интеграционное тестирование* — проверяет работоспособность компонентов системы на уровне взаимодействия нескольких отдельных исполняемых процессов. На данном этапе тестируется функционирование клиент-серверных систем, их взаимодействие внутри и с внешними компонентами.

*Функциональное тестирование* — рассматривает продукт, состоящий из множества классов, процессов, компонентов, данных как единое целое. На этом этапе проверяется в целом его работоспособность, функциональные и технические характеристики, а также бизнес-логика.

*Тестирование интерфейса* — проверка клиентских и административных интерфейсов пользователя на возможность выполнения с их помощью сценариев использования. Сценарий использования представляет собой последовательность действий пользователя, которые имитируют его активность при работе с интерфейсами системы. Сценарий использования должен покрывать спецификацию требований к пользовательскому интерфейсу. Такое тестирование производится в ручном и в автоматическом режиме с помощью специализированных утилит. Тестирование должно проверять корректность работы интерфейсной части приложения при любых возможных настройках экрана (различное разрешение, масштаб, шрифт), при изменениях фокуса, при работе с мышью и клавиатурой.

*Нагрузочное тестирование* — определение и проверка характеристик производительности системы в заданной конфигурации оборудования и набора данных.

*Тестирование базы данных* — проверка функционирования внешней базы данных и хранимых процедур в соответствии со спецификацией требований. Проверка политики безопасности доступа к базе в соответствии с ролями системы. Определение и проверка характеристик базы данных, таких как производительность, среднее время доступа, максимальное количество обслуживаемых клиентов, минимальная и максимальная длительность обработки запроса и т.п.

*Тестирование безопасности* — определение ролей и проверка списка функций системы, доступных для каждой роли. Может осуществляться на уровне интерфейса, на уровне компонента, на уровне базы данных, на уровне модуля и на сетевом уровне. Включает проверку методов шифрования данных при хранении и передаче, отказа доступа к запрещенным функциям, перехвата данных, подделки удостоверения личности, отказа в обслуживании и других атак.

*Тестирование конфигурации* — проверка работоспособности системы в заданном окружении конфигурации оборудования и набора данных.

*Регрессионное тестирование* — повторное выборочное тестирование продукта с модифицированными частями после исправления ошибки, добавления новой функции, рефакторинга и изменения кода. Внесение изменений в исходный код может повлечь цепочку зависимостей и получение новых ошибок во взаимозависимых функциях. Данный вид тестирования минимизирует риск подобного события.

*Тестирование инсталляции* — проверка корректной работы инсталляционного пакета, инсталляционных сценариев для копирования, обновления и последующей автоматической настройки работоспособности системы.

*Тестирование документации* — проверка документации на полноту описания инструкций пользования в соответствии с «Требованиями по разработке пакета рабочей документации пользователя и администратора системы».

**Инструментарий**

Для тестирования Продукта будут использованы следующие средства:

|  |  |  |
| --- | --- | --- |
|  | **Инструмент** | **Версия** |
| Инструмент хранения тест-кейсов | TestLink, MS Word | 2021 |
| Инструмент для функционального тестирования | *VS* | 2022 |
| Инструмент для тестирования производительности и нагрузки | *VS* | 2022 |
| Система отслеживания ошибок | *VS* | 2022 |
| Инструменты тестирования СУБД | *SQL Server Management Studio* | 2022 |
| Инструменты формирования отчётов | Word, Excel | 2021 |

**4. КРИТЕРИИ ТЕСТИРОВАНИЯ**

**Критерии успешности тестирования:**

* все тест кейсы с высоким приоритетом закрыты с результатом «пройден».
* тестовое покрытие проверено и является достаточным, где критерий достаточности составляет не менее 99% покрытия требований тестами;
* итоговый отчёт составлен и утвержден руководителем тестирования и заказчиком.

**Критерии прерывания и продолжения тестирования:**

* критерием прерывания тестирования является появления и занесения в систему отслеживания ошибок блокирующих ошибок;
* критерием продолжения тестирования является закрытие блокирующей ошибки в системе отслеживания ошибок.

**5. ИТОГОВЫЕ ОТЧЁТЫ**

После окончания тестирования формируются отчёты, в которых отображаются следующие данные:

* количество всех тестов;
* количество: всех тестов; пройденных успешно; не пройденных успешно; тестов, которые не выполнялись;
* информация о критических ошибках, размещённых в системе отслеживания ошибок;
* информация о времени, затраченном на тестирование;
* информация о дате начала и завершения тестирования;
* информация о количестве ошибок: всего ошибок; закрыто ошибок; новые ошибки выявленные в процессе тестирования; новые ошибки выявленные в процессе эксплуатации.
* информация о критических ошибках, размещённых в системе отслеживания ошибок и влияющих на готовность Продукта;
* Оценка степени готовности Продукта.

**Лабораторное занятие №4**

**Тема:** функциональное тестирование пользовательского интерфейса. Составление отчета об ошибках.

**Цели занятия:**

- освоить методы тестирования функций приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Функциональное тестирование.

Отчет о проведенном функциональном тестировании пользовательского интерфейса программы для учета информации о сотрудниках:

1) Ошибка: Некорректное размещение элементов управления на экранных формах.

Действие: при открытии формы "Добавить нового сотрудника" кнопка "Сохранить" расположена не в центре формы, что создает неудобство для пользователя.

Ожидаемый результат: Кнопка "Сохранить" должна быть центрирована на форме.

Приоритет: Средний

2) Ошибка: Некорректное оформление выводимых сообщений.

Действие: при успешном добавлении нового сотрудника выводится стандартное системное сообщение без дополнительной информации.

Ожидаемый результат: Вывод сообщения о добавлении нового сотрудника с указанием его имени и должности.

Приоритет: Низкий

3) Ошибка: Некорректный формат ввода данных.

Действие: при вводе даты рождения сотрудника допускается ввод некорректного формата (например, не числовые символы).

Ожидаемый результат: Ограничение на ввод только числовых символов в поле даты рождения.

Приоритет: Высокий

4) Ошибка: Некорректная реакция системы на ввод пользователя.

Действие: при попытке удаления сотрудника без подтверждения система выполняет операцию удаления без предупреждения пользователя.

Ожидаемый результат: Появление окна с запросом подтверждения удаления сотрудника перед выполнением операции.

Приоритет: Средний

5) Ошибка: Длительное время отклика на команды пользователя.

Действие: при выполнении операции поиска сотрудника система отвечает с задержкой более 5 секунд.

Ожидаемый результат: Отклик системы на команды пользователя должен быть менее 2 секунд.

Приоритет: Высокий

В ходе функционального тестирования были выявлены 5 ошибок, которые необходимо исправить для повышения удобства использования программы пользователем.

**Лабораторное занятие №5**

**Тема:** тестирование удобства использования пользовательского интерфейса. Составление отчета об ошибках.

**Цели занятия:**

- освоить методы тестирования функций приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Тестирование удобства использования пользовательского интерфейса.

После проведения тестирования удобства использования пользовательского интерфейса для учета информации о сотрудниках были выявлены следующие ошибки:

1.1. Наблюдаемость состояния системы:

- Оповещения системой пользователя о текущих действиях отсутствуют или отображаются не в удобный для восприятия способ.

1.2. Соотнесение с реальным миром:

- Терминология в интерфейсе не соответствует предметной области, что затрудняет понимание системы.

1.3. Пользовательское управление и свобода действий:

- Отсутствует возможность аварийного выхода, что не позволяет пользователям вернуться к предыдущему нормальному состоянию.

1.4. Целостность и стандарты:

- Использование различных слов для обозначения одних и тех же объектов и действий, что приводит к путанице у пользователей.

1.5. Помощь пользователям в распознавании, диагностике и устранении ошибок:

- Сообщения об ошибках написаны сложным языком и не содержат четкого определения проблемы и ее решения.

1.6. Предотвращение ошибок:

- Не предусмотрены элементы, предотвращающие возникновение ошибок у пользователя.

1.7. Распознавание, а не вспоминание:

- Интерфейс создает большую нагрузку на память пользователя из-за неясности, недоступности и неявной видимости объектов и действий.

1.8. Гибкость и эффективность использования:

- Отсутствуют горячие клавиши и возможность автоматизации часто повторяющихся действий.

1.9. Эстетичный и минимально необходимый дизайн:

- Присутствует избыточная информация, которая не относится к основной задаче пользователей.

1.10. Помощь и документация:

- Система помощи неудобна в использовании и не содержит пошаговых руководств по выполнению задач.

Исходя из выявленных ошибок, рекомендуется провести переработку пользовательского интерфейса с учетом указанных характеристик для повышения удобства использования системы.

**Лабораторное занятие №6**

**Тема:** системное тестирование.

**Цели занятия:**

- освоить методы тестирования функций приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Системное тестирование.

Отчет о результате системного тестирования программы для учета информации о сотрудниках:

1. Программа успешно загружается без ошибок.

2. Основные функции программы работают правильно: ввод данных, обработка данных, взаимодействие с пользователем и вывод результатов.

3. Программа была протестирована на различных наборах тестовых данных, включая корректные и некорректные значения. Все тесты прошли успешно.

4. Результаты работы программы соответствуют ожидаемым значениям на всех этапах выполнения.

5. Все возможные пути выполнения программы были протестированы, включая различные варианты взаимодействия пользователя с программой.

6. Производительность программы была проверена, время выполнения основных операций приемлемо, ресурсы компьютера используются эффективно.

7. Программа совместима с различными операционными системами и конфигурациями оборудования.

8. Обработка ошибок и исключительных ситуаций в программе работает корректно.

Обнаруженных ошибок не было. Рекомендации по улучшению программы:

- Добавить возможность экспорта и импорта данных о сотрудниках для удобного обмена информацией.

- Расширить функционал программы, добавив возможность создания отчетов и статистики по сотрудникам.

Общая оценка качества программы: высокое качество, все функции работают корректно, программу можно рекомендовать для использования.

**Лабораторное занятие №7**

**Тема:** модульное тестирование.

**Цели занятия:**

- освоить методы тестирования функций приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Модульное тестирование.

**Лабораторное занятие №8**

**Тема:** ручное тестирование.

**Цели занятия:**

- Освоить методы ручного тестирования приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Ручное тестирование.

Для проведения ручного тестирования программы для учёта информации о сотрудниках был использован следующий набор тестов:

1. Проверка возможности добавления нового сотрудника.

2. Проверка возможности редактирования информации о сотруднике.

3. Проверка возможности удаления сотрудника из базы данных.

4. Проверка корректности отображения списка всех сотрудников.

5. Проверка поиска сотрудника по имени или фамилии.

В результате тестирования были выявлены следующие ошибки:

1. При добавлении нового сотрудника возможно сохранение записи с пустыми обязательными полями.

2. При редактировании информации о сотруднике нет возможности изменить его должность или отдел, что ограничивает функционал программы.

3. При удалении сотрудника не происходит удаление его данных из всех связанных таблиц, что может привести к ошибкам в дальнейшем использовании программы.

Рекомендации по исправлению ошибок:

1. Добавить проверку на заполненность всех обязательных полей при добавлении нового сотрудника.

2. Добавить функционал изменения должности и отдела сотрудника при редактировании информации.

3. Добавить механизм удаления данных о сотруднике из всех связанных таблиц при удалении его из базы данных.

Общая оценка качества программы: 7 из 10. Программа имеет базовый функционал для учёта информации о сотрудниках, однако требует доработки и исправления выявленных ошибок для повышения её эффективности и удобства использования.

**Лабораторное занятие №9**

**Тема:** интеграционное тестирование.

**Цели занятия:**

- освоить методы интеграционного тестирования приложения;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Интеграционное тестирование.

Интеграционное тестирование программы для учёта информации о сотрудниках прошло успешно. Было проведено добавление данных на одной странице (страница с информацией о сотруднике) и изменение данных на другой странице (страница с редактированием информации о сотруднике). Проверка корректности передачи данных между страницами также была успешно завершена.

В ходе тестирования были обнаружены следующие ошибки:

1. Некорректное отображение даты рождения сотрудника на странице с информацией о сотруднике.

2. Ошибка при сохранении изменений после редактирования информации о сотруднике.

Рекомендации по исправлению ошибок:

1. Исправить алгоритм отображения даты рождения сотрудника на странице с информацией о нем.

2. Проверить и исправить ошибку при сохранении изменений после редактирования информации о сотруднике.

Общая оценка качества программы: программа имеет хороший функционал для учёта информации о сотрудниках, однако необходимо провести доработки для исправления обнаруженных ошибок. После внесения улучшений программа будет готова к использованию в реальных условиях.

**Лабораторное занятие №10**

**Тема:** оформление отчета проверки качества.

**Цели занятия:**

- Освоить составление итогового отчета по результатам тестирования;

- развитие логического мышления, самостоятельности в принятии технологии и методов выполнения задания.

**Оборудование:** ПК, ЛВС.

**Ход работы:**

Задание 1. Отчёт проверки качества созданного приложения.