Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Технологии разработки программного обеспечения

Отчёт
по лабораторной работе №7
на тему
Тестирование ПП и экономическое обоснование его реализации

Выполнил:
студент гр. 053502
Макаро М. В.

Проверил:
Ассистент кафедры информатики
Гриценко Н. Ю.

Минск 2023

СОДЕРЖАНИЕ

1 ПОСТАНОВКА ЗАДАЧИ

1 Разработать план тестирования (определение тестовых случаев и тестовых данных).

2 Реализация тестов: тесты на прием и обработку входных данных, тесты на корректность выполнения основных функций ПП, тесты на обработку и вывод результатов.

3 Провести тестирование, анализ тестирования и документирование найденных ошибок (если таковые имеются) с последующим исправлением их.

4 Составить экономическое обоснование разработки ПП.

5 Подготовить отчет по результатам выполнения лабораторной работы.

2 РАЗРАБОТКА ПЛАНА ТЕСТИРОВАНИЯ

План тестирования проиллюстрирован в таблице 2.1

Таблица 2.1 – План тестирования

Тестовый случай	Тестовые данные	Ожидаемый результат
Ввод корректного логина	Корректный логин	Проверка пройдена
Ввод логина с тире	Логин с тире	Проверка пройдена
Ввод логина с запрещенными символами	Логин с запрещенными символами	Проверка не пройдена
Ввод пустого логина	Пустой логин	Проверка не пройдена
Ввод очень длинного логина	Очень длинный логин	Проверка не пройдена
Ввод корректного пароля	Корректный пароль	Проверка пройдена
Ввод пароля с запрещенными символами	Пароль с запрещенными символами	Проверка не пройдена
Ввод короткого пароля	Короткий пароль	Проверка не пройдена
Ввод очень простого пароля	Простой пароль	Проверка не пройдена
Ввод пустого пароля	Пустой пароль	Проверка не пройдена
Ввод очень длинного пароля	Длинный пароль	Проверка не пройдена
Ввод текста в поле, для чисел	Текст	Пустой поле
Ввод неверного логина	Неверный логи	Появление оповещения
Ввод неверного пароля	Неверный пароль	Появление оповещения
Сортировка пользователей по производительности	Список сотрудников	Список отсортированный по производительности
Нахождение производительности работника	Работник	Его производительность
Отображение строки в отчете	Отчет	Корректное отображение информации
Сохранение пустого отчета	Пустой отчет	Оповещение об ошибке

Продолжение таблицы 2.1

Тестовый случай	Тестовые данные	Ожидаемый результат
Отображение сотрудника в списке	Информация о сотруднике	Корректное отображение информации
Отображение информации о задании	Информация о задании	Корректное отображение информации

План тестирования программного продукта (ПП) – это документ, который определяет, как, что и когда будет тестироваться в ПП. Этот процесс включает определение целей, аудитории, тестовых случаев, приоритетов, рисков, методологии, инструментов, ресурсов и сроков. Затем проводятся тесты, а результаты анализируются с созданием отчета. План тестирования помогает обеспечить качество ПП и удостовериться, что оно соответствует ожиданиям пользователей.

3 РЕАЛИЗАЦИЯ ТЕСТОВ

3.1 Тесты на прием и обработку входных данных

Тесты на прием и обработку входных данных являются одним из видов модульного тестирования, предназначенного для проверки правильности обработки входных данных программой. Эти тесты сосредотачиваются на том, как программа взаимодействует с входными данными, какие данные она принимает, как их обрабатывает и какие результаты выдает в ответ.

Вот некоторые ключевые аспекты тестов на прием и обработку входных данных:

1 Проверка корректности форматов данных: тесты проверяют, что программа правильно обрабатывает данные в соответствии с их форматом, например, числовые данные, строки, даты, времена и т. Д. Если формат данных неверен, программа должна либо отклонить их, либо преобразовать в корректный формат.

2 Проверка обработки нулевых и пустых значений: тесты могут включать в себя проверку того, как программа обрабатывает нулевые значения (если они допустимы) и пустые строки.

3 Проверка обработки исключительных ситуаций: тесты должны включать в себя ситуации, когда входные данные не соответствуют ожиданиям, и программа должна правильно обрабатывать исключения или ошибки.

4 Проверка диапазона данных: если программа должна обрабатывать данные в определенном диапазоне (например, только положительные числа), тесты должны убедиться, что она это делает корректно.

5 Проверка взаимодействия с внешними данными: в случаях, когда программа взаимодействует с внешними источниками данных (например, базами данных, API), тесты могут проверять корректность запросов и обработку возвращаемых данных.

6 Тестирование ввода через интерфейс пользователя: если программа имеет пользовательский интерфейс, тесты могут включать в себя сценарии взаимодействия пользователя с приложением, чтобы убедиться, что ввод обрабатывается правильно.

7 Создание реалистичных тестовых данных: важно создавать тестовые данные, которые максимально приближены к реальным сценариям использования приложения.

8 Автоматизация и изоляция: тесты на прием и обработку входных данных легко автоматизировать и должны выполняться в изолированной среде, чтобы исключить внешние факторы.

Тестирование на прием и обработку входных данных помогает обнаружить ошибки, связанные с некорректными или некорректно обработанными данными, и убедиться в корректности работы программы в различных сценариях ввода.

В таблице 3.1 показаны наименования классов и тестов с рассматриваемым тестовым случаем. Исходный код предоставлен в Приложении А.

Таблица 3.1 – Тесты на приём и обработку входных данных

Именование класса	Именование тестов	Тестовый случай
AuthorizationUtilsTests	CorrectLoginTest	Ввод корректного логина
	LoginWithDashTest	Ввод логина с тире
	LoginWithIncorrectSymbolTest	Ввод логина с запрещенными символами
	EmptyLoginTest	Ввод пустого логина
	VeryLongLoginTest	Ввод очень длинного логина
	CorrectPasswordTest	Ввод корректного пароля
	PasswordWithIncorrectSymbolTest	Ввод пароля с запрещенными символами
	ShortPasswordTest	Ввод короткого пароля
	VerySimplePasswordTest	Ввод очень простого пароля
	EmptyPasswordTest	Ввод пустого пароля
	VeryLongPasswordTest	Ввод очень длинного пароля
CreateReportWidgetTests	InputTextInNumFieldTest	Ввод текста в числовое поле
AuthorizationWidgetTests	TestInputIncorrectLogin	Ввод неверного логина и проверка оповещения
	TestInputIncorrectPassword	Ввод неверного пароля и проверка оповещения

3.2 Тесты на корректность выполнения основных функций ПП

Тесты на корректность выполнения основных функций программного продукта предназначены для проверки того, что ключевые функции и алгоритмы программы работают правильно и выполняют свою основную задачу в

соответствии с заявленными требованиями. Эти тесты фокусируются на проверке правильности работы отдельных частей кода и их взаимодействия друг с другом.

Вот некоторые ключевые аспекты тестов на корректность выполнения основных функций программы:

1 Тестирование основных алгоритмов: тесты должны проверять корректность выполнения ключевых алгоритмов, которые обеспечивают основную функциональность программы.

2 Тестирование правильности решения задачи: тесты должны проверять, что программа действительно решает основную задачу, для которой она предназначена.

3 Проверка обработки разных случаев: тесты должны включать в себя разнообразные сценарии использования программы, включая различные варианты входных данных, краевые случаи и исключительные ситуации. Это позволяет убедиться, что программа правильно обрабатывает разнообразные сценарии [1].

4 Тестирование обработки ошибок: в случае возникновения ошибок или исключений, программа должна правильно обрабатывать ситуацию, например, возвращать информативные сообщения об ошибках.

5 Тестирование производительности: в некоторых случаях важно проверить, что основные функции выполняются в приемлемые сроки и не вызывают проблем с производительностью.

6 Изоляция тестируемых частей кода: тесты на корректность выполнения основных функций должны быть изолированы от других частей программы и выполняться независимо.

7 Автоматизация: тесты на корректность выполнения основных функций легко автоматизировать и многократно выполнять в ходе разработки и поддержки продукта.

Эти тесты помогают обнаруживать ошибки и проблемы в ключевых частях программы, обеспечивая ее надежность и корректную работу. Тесты на корректность выполнения основных функций являются важным элементом тестирования, так как именно они проверяют, что программа выполняет свою основную задачу в соответствии с требованиями и ожиданиями пользователей.

В таблице 3.2 показаны наименования классов и тестов с рассматриваемым тестовым случаем.

Таблица 3.2 – Тесты на корректность выполнения основных функций ПП

Именование класса	Именование тестов	Тестовый случай
BllTests	SortByProductivity-Test	Сортировка по продуктивности
	CalcProductivityTest	Вычисление продуктивности

3.3 Тесты на обработку и вывод результатов

Тесты на обработку и вывод результатов (output and results testing) предназначены для проверки правильности обработки данных программой и корректного вывода результата или информации пользователю. Эти тесты оценивают, насколько хорошо программа взаимодействует с внешним миром и предоставляет пользователю ожидаемые результаты.

Некоторые ключевые аспекты тестов на обработку и вывод результатов:

1 Тестирование вывода результатов: этот тип тестов проверяет, что программа корректно форматирует и выводит результат своей работы. Это может включать в себя проверку правильности вывода текстовых сообщений, генерации отчетов, создания файлов и других способов предоставления информации пользователю.

2 Тестирование интерфейса пользователя: если программа имеет графический интерфейс (GUI), тесты проверяют правильность отображения элементов интерфейса, их доступность и функциональность. Это также включает в себя тестирование навигации, взаимодействия с элементами управления и общего пользовательского опыта.

3 Проверка взаимодействия с внешними системами: если программа взаимодействует с внешними системами, такими как базы данных, веб-сервисы или другие приложения, тесты должны проверять правильность этого взаимодействия, включая передачу данных, получение ответов и обработку ошибок.

4 Тестирование обработки ошибок и исключений: эти тесты оценивают, как программа обрабатывает ситуации ошибок и исключений, включая вывод информативных сообщений об ошибках, корректное завершение программы и предотвращение возможных сбоев.

5 Проверка правильности данных на выходе: эти тесты оценивают, что данные или результаты, полученные от программы, соответствуют ожиданиям. Например, проверка правильности значений в выходных файлах, соответствие расчетов заранее определенным эталонам и т.д.

6 Тестирование производительности вывода: некоторые тесты могут оценивать скорость и эффективность вывода результатов, особенно в случае больших объемов данных или сложных отчетов.

7 Интеграция с автоматизированными системами тестирования: для продуктов, взаимодействующих с другими системами, важно иметь тесты, которые могут быть легко интегрированы в автоматизированные системы непрерывной интеграции (CI) и доставки (CD) [2].

8 Тестирование на разных платформах и окружениях: если программа должна работать на разных платформах, браузерах, устройствах или окружениях, тесты должны учитывать разнообразие сценариев использования.

Тесты на обработку и вывод результатов помогают удостовериться, что программа предоставляет пользователю точные, информативные и полезные результаты и взаимодействует с внешним миром правильным образом. Они

также могут обнаруживать проблемы, связанные с пользовательским интерфейсом, производительностью и обработкой ошибок, что важно для обеспечения качества программного продукта.

В таблице 3.3 показаны наименования классов и тестов с рассматриваемым тестовым случаем.

Таблица 3.3 – Тесты на обработку и вывод результатов

Именование класса	Именование тестов	Тестовый случай
WidgetsOutputTests	OutputReportStrTest	Проверяет вывод строки в отчете
	SaveEmptyReportTest	Проверка сохранения пустого отчета
	OutputEmployeeInfoTest	Проверка вывода информации о сотрудника
	OutputTaskInfoTest	Проверка вывода информации о задаче

4 ПРОВЕДЕНИЕ ТЕСТИРОВАНИЯ

На рисунках 4.1 и 4.2 показан результат работы двадцати тестов, распределённых по классам в зависимости от выполняемой задачи.

```
********* Start testing of AuthorizationUtilsTests *********
Config: Using QtTest library 5.15.2, Qt 5.15.2 (x86_64-little_endian-llp64 shared (dynamic) release build; by GCC 8.1.0), windows 10
PASS   : AuthorizationUtilsTests::initTestCase()
PASS   : AuthorizationUtilsTests::IsLoginCorrectTest(CorrectLoginTest)
PASS   : AuthorizationUtilsTests::IsLoginCorrectTest(LoginWithDashTest)
PASS   : AuthorizationUtilsTests::IsLoginCorrectTest(LoginWithIncorrectSymbolTest)
PASS   : AuthorizationUtilsTests::IsLoginCorrectTest(EmptyLoginTest)
PASS   : AuthorizationUtilsTests::IsLoginCorrectTest(VeryLongLoginTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(CorrectPasswordTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(PasswordWithIncorrectSymbolTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(ShortPasswordTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(VerySimplePasswordTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(EmptyPasswordTest)
PASS   : AuthorizationUtilsTests::IsPasswordCorrectTest(VeryLongPasswordTest)
PASS   : AuthorizationUtilsTests::cleanupTestCase()
Totals: 13 passed, 0 failed, 0 skipped, 0 blacklisted, 2ms
********* Finished testing of AuthorizationUtilsTests *********
********* Start testing of AuthorizationWidgetTests *********
Config: Using QtTest library 5.15.2, Qt 5.15.2 (x86_64-little_endian-llp64 shared (dynamic) release build; by GCC 8.1.0), windows 10
PASS   : AuthorizationWidgetTests::initTestCase()
PASS   : AuthorizationWidgetTests::TestInputIncorrectLogin()
PASS   : AuthorizationWidgetTests::TestInputIncorrectPassword()
PASS   : AuthorizationWidgetTests::cleanupTestCase()
Totals: 4 passed, 0 failed, 0 skipped, 0 blacklisted, 1ms
********* Finished testing of AuthorizationWidgetTests *********
********* Start testing of BllTests *********
Config: Using QtTest library 5.15.2, Qt 5.15.2 (x86_64-little_endian-llp64 shared (dynamic) release build; by GCC 8.1.0), windows 10
PASS   : BllTests::initTestCase()
PASS   : BllTests::SortByProductivityTest()
PASS   : BllTests::CalcProductivityTest()
PASS   : BllTests::cleanupTestCase()
Totals: 4 passed, 0 failed, 0 skipped, 0 blacklisted, 1ms
```

Рисунок 4.1 – Первая часть результата работы тестов

```
********* Start testing of CreateReportWidgetTests *********
Config: Using QtTest library 5.15.2, Qt 5.15.2 (x86_64-little_endian-llp64 shared (dynamic) release build; by GCC 8.1.0), windows 10
PASS   : CreateReportWidgetTests::initTestCase()
PASS   : CreateReportWidgetTests::InputTextInNumFieldTest()
PASS   : CreateReportWidgetTests::cleanupTestCase()
Totals: 3 passed, 0 failed, 0 skipped, 0 blacklisted, 1ms
********* Finished testing of CreateReportWidgetTests *********
********* Start testing of WidgetsOutputTests *********
Config: Using QtTest library 5.15.2, Qt 5.15.2 (x86_64-little_endian-llp64 shared (dynamic) release build; by GCC 8.1.0), windows 10
PASS   : WidgetsOutputTests::initTestCase()
PASS   : WidgetsOutputTests::OutputReportStrTest()
PASS   : WidgetsOutputTests::SaveEmptyReportTest()
PASS   : WidgetsOutputTests::OutputEmployeeInfoTest()
PASS   : WidgetsOutputTests::OutputTaskInfoTest()
PASS   : WidgetsOutputTests::cleanupTestCase()
Totals: 6 passed, 0 failed, 0 skipped, 0 blacklisted, 1ms
********* Finished testing of WidgetsOutputTests *********
```

Рисунок 4.2 –Вторая часть результата работы тестов

5 ТЕХНИКО–ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ

5.1 Характеристика программного средства, разрабатываемого для реализации на рынке

EfficiencyForge – это новый продукт, который представляет собой программное обеспечение для повышения эффективности работы команд, отдельных отделов и предприятий в целом. Данный программный продукт создается для решения изложенных ниже задач.

Помочь рядовому сотруднику четко понимать проекты, над которыми он работает, и конкретные задачи в рамках этих проектов. Также он поможет наглядно показать сотруднику, сколько времени он по итогу тратит на разные задачи, или показать, что большую часть рабочего времени он бездельничает.

Сделать процесс управления рабочим временем более эффективным для руководителей, позволяя им быстро анализировать данные о рабочем времени подчиненных сотрудников и принимать необходимые решения на основе этих данных.

Предоставить возможность анализа данных о рабочем времени с целью повышения эффективности бизнеса, оптимизации рабочих процессов и улучшения ресурсного планирования.

Сократить затраты на административные процессы, связанные с учетом рабочего времени, благодаря автоматизации и оптимизации задач.

Сократить время, необходимое на такие рутинные дела, как согласование отпуска и оповещение о командировках.

Так же EfficiencyForge является хорошо оптимизированным и кроссплатформенным решением, которое с очень большой вероятностью можно внедрить в уже существующий у организаций аппаратный комплекс. Еще одним плюсом данного решения является тот факт, что все данные будут храниться внутри организации и передаваться только по локальной сети. Нет необходимости отправлять потенциально важные коммерческие данные на чей-то удаленный сервер, что повышает безопасность данного решения.

В заключение, EfficiencyForge – это отличный выбор для малых и средних предприятий, которым нужен простой и доступный продукт без сложной инфраструктуры. EfficiencyForge предлагает необходимые инструменты для повышения эффективности организации.

5.2 Расчет затрат на разработку и реализацию

Разработку программного продукта будет осуществлять команда разработчиков в составе:

– UI/UX дизайнер. Участвует в разработке внешнего вида ПП, занимаясь его оформлением.

– C++ разработчики, занимающиеся разработкой;

– Специалист. Занимается продвижением ПП и технической поддержкой.

Каждый из специалистов команды разработчиков работает над своей частью проекта и трудоемкость выполнения работ составляет 20 ч. И 160 ч. Соответственно. Специалист занимается сопровождением программного продукта после его разработки. Размеры окладов специалистов компании и расчет основной заработной платы разработчиков представлены в таблице 5.1.

Таблица 5.1 – Расчет затрат на основную заработную плату команды разработчиков

Категория исполнителей	Месячный оклад, р.	Часовой оклад, р.	Трудоемкость работ, ч	Итого, р.
UI/UX дизайнер	1200	7,5	20	150
C++ разработчик	1600	10	160	1600
Специалист	1600	10	1760	17600
Всего затраты на основную заработную плату				19350

Общая сумма затрат на разработку продукта рассчитывается по методике расчета цены программного средства, представленной в таблице 5.2.

Таблица 5.2 – Расчёт затрат на разработку программного средства

Наименование статьи затрат	Формула/таблица для расчетов	Значение, р.
Основная заработная плата разработчиков ($З_о$)	Таблица 5.1	19350
Дополнительная заработная плата разработчиков ($З_д$)	$$З_д = \frac{З_о \cdot Н_д}{100} = \frac{19350 \cdot 10}{100}, \quad (5.1)$$ где $Н_д$ – норматив дополнительной заработной платы (10%)	1935
Отчисления на социальные нужды ($Р_{соц}$)	$$Р_{соц} = \frac{(З_о + З_д) \cdot Н_{соц}}{100} = \frac{(19350 + 1935) \cdot 34{,}6}{100}, \quad (5.2)$$ где $Н_{соц}$ – норматив отчислений в ФСЗН и Белгосстрах – 34,6%	7364,61

Продолжение таблицы 5.2

Прочие расходы ($Р_{пр}$)	$$Р_{пр} = \frac{З_о \cdot Н_{пр}}{100} = \frac{19350 \cdot 30}{100}, \quad (5.3)$$ где $Н_{пр}$ – норматив прочих расходов (30%)	5805
Расходы на реализацию ($Р_р$)	$$Р_р = \frac{З_о \cdot Н_р}{100} = \frac{19350 \cdot 3}{100}, \quad (5.4)$$ где $Н_р$ – норматив расходов на реализацию (3%)	580,5
Общая сумма затрат на разработку и реализацию	$$З_р = З_о + З_д + Р_{соц} + Р_{пр} + Р_р. \quad (5.5)$$	35035,11

Таким образом, в результате расчетов, представленных в таблице 5.2, сумма затрат на разработку программного средства составит 35035 рублей 11 копеек.

5.3 Расчет экономической эффективности от реализации проекта

Целевой аудиторией разрабатываемого программного продукта являются малые и средние предприятия, которым нужен простой и доступный продукт без сложной инфраструктуры.

Экономический эффект организации-разработчика программного продукта представляет собой получение прибыли путем продажи лицензии на определенный срок.

На основе опроса знакомых руководителей организаций можно сделать вывод, что они видят много плюсов в такого рода программных продуктах, и согласны с необходимостью внедрения такого программного обеспечения в организациях. Таким образом, на основании данного опроса, можно предположить, что удастся продать около 35 лицензий на 12 месяцев.

Так же на основе опроса, упомянутого выше, и анализа цен на аналогичные программные средства, представленные на рынке, цена за лицензию на один персональный компьютер на один месяц будет составлять 4 рубля.

Налог на добавленную стоимость определяется по формуле ниже:

$$НДС = \frac{Ц_{отп} \cdot N \cdot Н_{д.с.}}{100\% + Н_{д.с.}} \quad (5.6)$$

где $Н_{д.с}$ – ставка налога на добавленную стоимость (20%); $Ц_{отп}$ – отпускная цена копии программного средства в белорусских рублях; N – количество копий, реализуемых за год.

Рассчитаем НДС по формуле 5.7.

$$НДС = \frac{4 \cdot 12 \cdot 35 \cdot 20}{100\% + 20\%} = 280 \text{ руб}$$

Прирост чистой прибыли, полученной разработчиком от реализации программного средства на рынке рассчитаем по формуле:

$$\Delta П_{ч}^{р} = (Ц_{отп} \cdot N - НДС) \cdot Р_{пр} \left(1 - \frac{Н_{п}}{100}\right) \quad (5.7)$$

где $Ц_{отп}$ – отпускная цена копии программного средства в белорусских рублях; N – количество копий, реализуемых за год; НДС – сумма налога на добавленную стоимость в белорусских рублях; $Р_{пр}$ – рентабельность продаж копий в процентах (40%); $Н_{п}$ – ставка налога на прибыль в процентах (20%).

Рассчитаем прирост чистой прибыли по формуле 5.7.

$$\Delta П_{ч}^{р} = (4 \cdot 12 \cdot 35 - 280) \cdot 40 \cdot \left(1 - \frac{20}{100}\right) = 44800 \text{ руб}$$

5.4 Расчет показателей экономической эффективности

Оценку экономической эффективности инвестиций в разработку программного средства осуществим с помощью расчета рентабельности инвестиций (Return on Investment, ROI) по следующей формуле:

$$ROI = \frac{\Delta П_{ч}^{р} - З_{р}}{З_{р}} \cdot 100\% \quad (5.8)$$

где $\Delta П_{ч}^{р}$ – прирост чистой прибыли в белорусских рублях; $З_{р}$- затраты на разработку программного средства в белорусских рублях.

Рассчитаем рентабельность инвестиций по формуле 5.8.

$$ROI = \frac{44800 - 35035{,}11}{35035{,}11} \cdot 100\% = 27{,}87\%$$

Ставка по долгосрочным депозитам для юридических лиц составляет 8,5 процентов. Исходя из того, что, показатель ROI превышает ставку по долгосрочным депозитам, можно сделать вывод о целесообразности разработки и реализации программного средства.

ЗАКЛЮЧЕНИЕ

В ходе выполнения лабораторной работы были разработаны план тестирования программного продукта, а также составлено экономическое обоснование его разработки.

В план тестирования входят тесты на входные данные, на функционал ПП, а также на обработку и вывод результатов.

Также была произведена оценка экономической эффективности программного продукта для планирования. Вычисленное значение в 27,87 процента показывает хорошую перспективу для его финансирования.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Википедия – Unit testing [Электронный ресурс]. – Режим доступа : https://en.wikipedia.org/wiki/UnitTesting.

[2] Google AdMob [Электронный ресурс]. – Режим доступа : https://admob.google.com/home/home-calc/.