**ПРАКТИЧЕСКАЯ РАБОТА №8**

**Тема:** Тестирование программного продукта методом «Белого ящика»  
**Цель:** Научиться использовать метод «белого ящика» для тестирования программ.

**1. Описание программы**

Программа предназначена для управления музыкальными группами, их песнями и гастролями.  
Основные функции:

* добавление групп, песен и гастролей;
* удаление песен;
* поиск песен и групп по различным критериям;
* вывод информации о гастролях, певцах, композиторах и т.д.

**2. Алгоритм (пример блок-схемы)**

Основной алгоритм (меню программы):

┌──────────────────────────┐

│ Запуск программы │

└────────────┬─────────────┘

│

┌───────▼────────┐

│ Вывести меню │

└───────┬────────┘

│

┌───────▼────────┐

│ Считать выбор │

└───────┬────────┘

│

┌───────────▼──────────────────────┐

│ 1 – AddGroup() │

│ 2 – AddSong() │

│ 3 – AddTour() │

│ ... │

│ 11 – SaveData() и выход │

└──────────────────────────────────┘

**3. Фрагмент программы для тестирования**

Для демонстрации метода «белого ящика» возьмем функцию **AddGroup()**, так как она содержит ветвления и проверки.

|  |
| --- |
| static void AddGroup()  {  Console.WriteLine("Название:");  var name = Console.ReadLine();  int year;  Console.WriteLine("Год:");  if (!int.TryParse(Console.ReadLine(), out year))  {  Console.WriteLine("Неверный формат года.");  return;  }  Console.WriteLine("Страна:");  var country = Console.ReadLine();  int chart;  Console.WriteLine("Позиция в чарте:");  if (!int.TryParse(Console.ReadLine(), out chart))  {  Console.WriteLine("Неверный формат позиции.");  return;  }  groups.Add(new Group(groups.Count + 1, name, year, country, chart));  Console.WriteLine("Группа добавлена.");  } |

**4. Потоковый граф (Flow Graph)**

|  |  |
| --- | --- |
| **Узел** | **Описание** |
| **1** | Ввод названия |
| **2** | Ввод года |
| **3** | Проверка int.TryParse(year) |
| **4** | Ошибка формата года |
| **5** | Ввод страны |
| **6** | Ввод позиции |
| **7** | Проверка int.TryParse(chart) |
| **8** | Ошибка формата позиции |
| **9** | Добавление группы и сообщение об успехе |

**Потоки управления:**

1 → 2 → 3 → 4 (если ошибка года)  
1 → 2 → 3 → 5 → 6 → 7 → 8 (если ошибка позиции)  
1 → 2 → 3 → 5 → 6 → 7 → 9 (успешное добавление)

**5. Цикломатическая сложность**

Формула:  
V(G) = E − N + 2,  
где  
E – количество дуг (переходов),  
N – количество узлов.

Из графа:  
E = 9, N = 8  
V(G) = 9 − 8 + 2 = 3

→ значит, достаточно 3 независимых тестовых пути.

**6. Независимые пути (базовое множество)**

| **№ пути** | **Путь** | **Описание** |
| --- | --- | --- |
| 1 | 1–2–3–4 | Ошибка при вводе года |
| 2 | 1–2–3–5–6–7–8 | Ошибка при вводе позиции |
| 3 | 1–2–3–5–6–7–9 | Успешное добавление |

**7. Критерии тестирования**

| **Критерий** | **Суть проверки** | **Реализовано** |
| --- | --- | --- |
| Покрытие операторов | Каждый оператор программы выполняется хотя бы один раз | + |
| Покрытие решений | Все ветви условий (TryParse = True/False) проверены | + |
| Покрытие условий | Проверены оба результата каждого условия | + |
| Покрытие маршрутов | Пройдены все 3 независимых пути | + |

**8. Таблица тестов**

| **№ теста** | **Назначение** | **Исходные данные** | **Ожидаемый результат** | **Фактический результат** |
| --- | --- | --- | --- | --- |
| **1** | Проверка неверного формата года | Назв: “Queen”, Год: “abc” | Сообщение: “Неверный формат года.” | Совпадает |
| **2** | Проверка неверного формата позиции | Назв: “ABBA”, Год: “1972”, Страна: “Швеция”, Позиция: “5” | Сообщение: “Неверный формат позиции.” | Совпадает |
| **3** | Проверка успешного добавления | Назв: “Beatles”, Год: “1960”, Страна: “UK”, Позиция: “1” | Сообщение: “Группа добавлена.” | Совпадает |

**9. Вывод**

Метод «белого ящика» позволил проверить:

* корректность всех ветвлений (TryParse());
* выполнение всех операторов программы;
* корректность сообщений при ошибках.

Все тесты пройдены успешно.  
Ошибок в логике функции не обнаружено.

**10. Контрольные вопросы**

**1. Почему стратегия называется «белый ящик»?**  
Потому что тестировщик имеет доступ к внутреннему коду программы и анализирует структуру логики.

**2. Что показывает цикломатическая сложность?**  
Количество независимых путей в программе, которое определяет минимальное число тестов для полного покрытия.

**3. В чем отличие покрытия условий и покрытия решений/условий?**  
Покрытие условий проверяет результаты каждого условия, а покрытия решений/условий — также все комбинации исходов условий в каждом решении.

**4. Какой критерий самый сильный?**  
Комбинаторное покрытие условий — оно проверяет все комбинации истинности условий.

**5. Порядок тестирования вложенных циклов:**

1. Сначала тестируется внутренний цикл;
2. Устанавливаются минимальные значения внешних циклов;
3. Затем проверяются внешние циклы с типовыми значениями вложенных;
4. Процесс повторяется до проверки всех уровней вложенности.