Отчет по лабораторной работе № 10 по курсу “Фундаментальная информатика”

Студент группы М80-109Б-22 Филиппов Александр Михайлович, № по списку 21

e-mail a.filipov04@yandex.ru
telegram @otaku0101

Работа выполнена: «7» апреля 2023г.

Преподаватель: каф. 806 Сысоев Максим Алексеевич

Отчет сдан « » _____20__ г., итоговая оценка _____

Подпись преподавателя _____

1. **Тема:** Отладчик системы программирования OS UNIX
2. **Цель работы:** Научиться пользоваться отладчиком
3. **Задание** Внесение и исрпавление ошибок различных типов
4. **Оборудование** (студента):
   Процессор *Intel Core i5-8265U @ 8x 3.9GH* с ОП *7851* Мб, НМД *1024* Гб. Монитор *1920x1080*
5. **Программное обеспечение (**студента**):**
   Операционная система семейства: *linux*, наименование: *ubuntu*, версия *18.10 cosmic*
   интерпретатор команд: *bash* версия *4.4.19.*
   Система программирования -- версия --**,** редактор текстов *emacs* версия *25.2.2*
   Утилиты операционной системы --
   Прикладные системы и программы --
   Местонахождение и имена файлов программ и данных на домашнем компьютере --

6. Идея, метод, алгоритм решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)

Генерируем различные виды ошибок, подробнее ниже

7. Сценарий выполнения работы [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

Входные данные	**Выходные данные**	**Описание тестируемого случая**
Ничего	Различные ошибки	Получаем шибки разных типов

8. Распечатка протокола (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

ОШИБКА RE #1 – использование неинициализированной переменной

```c
#include <stdio.h>

int main() {
    int i;

    for (int j = 0; j < i; ++j) {
        printf( format: "%d", i);
    }

    return 0;
}
```

РЕШЕНИЕ – инициализировать переменную

```c
#include <stdio.h>

int main() {
    int i;
    i = 1;
    for (int j = 0; j < i; ++j) {
        printf( format: "%d", i);
    }

    return 0;
}
```

ОШИБКА RE #2 – деление на ноль

```c
#include <stdio.h>

float Division(int a, int b) {
    float result = a / b;
    return result;
}

int main() {
    printf( format: "%f", Division( a: 10,  b: 0));
    return 0;
}
```

РЕШЕНИЕ – просто не делить на ноль

```c
#include <stdio.h>

float Division(float a, float b) {
    if (b == 0) {
        printf( format: "division by zero error");
        return b;
    } else {
        float result = a / b;
        return result;
    }
}

int main() {
    printf( format: "%f", Division( a: 10,  b: 0));
    return 0;
}
```

ОШИБКА CE #1 – деление массива на число

```c
#include <stdio.h>

int main() {
    int arr[2] = { [0]: 100, [1]: 100};
    int num = 10;

    printf( format: "%d", arr/num);

    return 0;
}
```

РЕШЕНИЕ – обратиться к числу по индексу

```c
#include <stdio.h>

int main() {
    int arr[2] = { [0]: 100, [1]: 100};
    int num = 10;

    printf( format: "%d", arr[0]/num);

    return 0;
}
```

ОШИБКА CE #2 - обращение к несуществующему полю

```c
int main() {
    int foo = 123;
    int bar = bar.field;

    printf("%d", bar);

    return 0;
}
```

РЕШЕНИЕ – создать объект с таким полем

```c
int main() {
    struct obj {
        int field;
    } foo;
    foo.field = 123;

    int bar = foo.field;

    printf("%d", bar);

    return 0;
}
```

ОШИБКА UB #1 - вызов функции, которая должна что-то вернуть, но не делает этого

```c
int foo() {
}

int maint() {
    return foo();
}
```

РЕШЕНИЕ – заставить функцию что-то возвращать

```c
int foo() {
    return 123;
}

int maint() {
    return foo();
}
```

ОШИБКА UB #2 - обращение к несуществующему элементу

```c
int main() {
    int t[123];

    printf("%d", t[321]);
    return 0;
}
```

РЕШЕНИЕ – не обращаться к несуществующему элементу

```c
int main() {
    int t[123];
    t[10] = 1;

    printf("%d", t[10]);
    return 0;
}
```

9. Дневник отладки должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

№	Лаб. или дом.	Дата	Время	Событие	Действие по исправлению	Примечание
0	дом	07.04.2023	17:00	Начал делать лабу		
1	дом	07.04.2023	18:30	Закончил делать лабу		

10. Замечания автора по существу работы

Отсутствуют.

11. Выводы

Не уверен насколько полезна данная ЛР, так как обычно ошибки сами тебя находят, и нужно их только исправлять. Возможно, такая практика поможет лучше понять причины каждой ошибки, позволяя в дальнейшем их быстрее фиксить, но на мой взгляд простые ошибки ты видишь и так, а сложные или раритетные гуглишь, поэтому и знать досконально каждую error нет большой необходимости.

Недочёты при выполнении задания могут быть устранены следующим образом: --

Подпись студента _____