**Содержание**

[1 Постановка задачи 2](#_Toc134714528)

[2 Заданный вариант 2](#_Toc134714529)

[3 Тексты скриптов 2](#_Toc134714530)

[4 Алгоритмы работы программы 36](#_Toc134714531)

[5 Топографическая карта 40](#_Toc134714532)

[6 Отрывки журналов работы 40](#_Toc134714533)

[Вывод 40](#_Toc134714534)

# 1 Постановка задачи

Основная задача данной курсовой работы – создать имитационную модель работы ВКО

# 2 Заданный вариант

Исходными данными были:

1. Координаты 3 РЛС, а также их ТТХ;
2. Координаты СПРО, дальность обнаружения, боезапас;
3. Координаты ЗРДН, дальность обнаружения, а также их боезапас;
4. Командный пункт

Рассмотрим вариант КР №4, который был задан:

РЛС1: координаты – город Донецк, a = 2700, тип – «Днепр»

РЛС2: координаты – x=8000, y=6000, a = 450, тип – «Дарьял»

РЛС3: координаты – x=8000, y=3500, a = 2700, тип – «Воронеж-ДМ»

ЗРДН1: координаты – город Петрозаводск, r = 600 км

ЗРДН2: координаты – город Пермь, r = 400 км

ЗРДН3: координаты – город Астана, r = 550 км

СПРО: координаты – город Минск, r = 1700 км

# 

# 3 Тексты скриптов

Приведу листинги программ, которые были разработаны в ходе выполнения данной курсовой работы:

Исходный код файла rls1.sh:

Файлы rls2.sh и rls3.sh имеют такую же логику и код, за исключением данных о самих РЛС, а именно: расположение, угол обзора и радиус обзора.

Исходный код файла zrdn1.sh:

Файлы zrdn2.sh и zrdn3.sh имеют такую же логику и код, за исключением данных о самих ЗРДН, а именно: расположение и радиус обзора.

Исходный код файла spro.sh:

Исходный код файла manage.sh:

# 4 Алгоритмы работы программы

Рассмотрим то, как работают приведённые выше листинги программ:

1. rls1.sh, rls2.sh и rls3.sh

В самом начале файла идет проверка на то, запущен ли этот файл уже или нет. Это необходимо для того, чтобы нельзя было повторно запустить работу РЛС.

После этого идет инициализация переменных и необходимой конфигурации станции. Дальше идет цикл «while», в начале которого идет отправка сообщения о своей работоспособности на КП. Далее идет считывание при каждой своей итерации последних добавленных файлов, которые генерируются от целей. Идет считывание новых координат. После идет проверка на то, попадают ли новые координаты в радиус работы РЛС. Если попадает, то идет проверка на то, попадала ли эта цель уже в радиус РЛС, если нет, то запоминаем эти координаты и ждем новые координаты, если да, то получаем старые координаты и осуществляем проверку на скорость цели. Если скорость подходит под нужную цель, а именно под БР, то идем обрабатывать цель дальше. Далее идет проверка на то, сообщали ли уже о этой цели, чтобы избежать выдачи повторной информации. Если нет, то сообщаем о том, что цель обнаружено в зоне действия РЛС. Далее производим проверку на то, летит ли цель в зону СПРО, путем вычисления прямой по двум координатам и определения пересечения между окружностью действия СПРО и дистанцией при первых координатах и вторых. Если да, то выводим необходимую информацию.

Вся выводимая информация передается на КП через шифрование по RSA, а также записывается в собственный «лог» файл.

Так же для РЛС 1 осуществляется дополнительная проверка, которая проверяет, находится ли цель уже в зоне СПРО, так как зона ее действия пересекается с зоной действия СПРО.

1. SPRO.sh

В самом начале файла идет проверка на то, запущен ли этот файл уже или нет. Это необходимо для того, чтобы нельзя было повторно запустить работу СПРО.

После этого идет инициализация переменных и необходимой конфигурации станции. Дальше идет цикл «while», в начале которого идет отправка сообщения о своей работоспособности на КП. Далее идет считывание при каждой своей итерации последних добавленных файлов, которые генерируются от целей. Идет считывание новых координат. После идет проверка на то, попадают ли новые координаты в радиус работы СПРО. Если попадает, то идет проверка на то, попадала ли эта цель уже в радиус СПРО, если нет, то запоминаем эти координаты и ждем новые координаты, если да, то получаем старые координаты и осуществляем проверку на скорость цели. Если скорость подходит под нужную цель, а именно под БР, то идем обрабатывать цель дальше. Далее идет проверка на то, сбивали ли уже эту БР. Если нет, то дальше идет проверка на то, сколько осталось противоракет, для того чтобы понимать, есть возможность сбивать или переходить на работу по обнаружению. Если ракеты есть, то осуществляем выстрел и выдаем об этом сообщение. Если проверка на то, сбивали ли уже выдает положительный результат, то идет проверка на то, сбили ли или нет. Если нет, то выдается информация о том, что цель не сбилась и необходим дополнительный выстрел. В конце итерации цикла «while» идет вывод информации о целях, которые удалось сбить.

Вся выводимая информация передается на КП через шифрование по RSA, а также записывается в собственный «лог» файл.

1. zrdn1.sh, zrdn2.sh и zrdn3.sh

В самом начале файла идет проверка на то, запущен ли этот файл уже или нет. Это необходимо для того, чтобы нельзя было повторно запустить работу ЗРДН.

После этого идет инициализация переменных и необходимой конфигурации станции. Дальше идет цикл «while», в начале которого идет отправка сообщения о своей работоспособности на КП. Далее идет считывание при каждой своей итерации последних добавленных файлов, которые генерируются от целей. Идет считывание новых координат. После идет проверка на то, попадают ли новые координаты в радиус работы ЗРДН. Если попадает, то идет проверка на то, попадала ли эта цель уже в радиус ЗРДН, если нет, то запоминаем эти координаты и ждем новые координаты, если да, то получаем старые координаты и осуществляем проверку на скорость цели. Если скорость подходит под нужную цель, а именно под Самолеты и Крылатые ракеты, то идем обрабатывать цель дальше. Далее идет проверка на то, сбивали ли уже эту цель. Если нет, то дальше идет проверка на то, сколько осталось противоракет, для того чтобы понимать, есть возможность сбивать или переходить на работу по обнаружению. Если ракеты есть, то осуществляем выстрел и выдаем об этом сообщение. Если проверка на то, сбивали ли уже выдает положительный результат, то идет проверка на то, сбили ли или нет. Если нет, то выдается информация о том, что цель не сбилась и необходим дополнительный выстрел. В конце итерации цикла «while» идет вывод информации о целях, которые удалось сбить.

Вся выводимая информация передается на КП через шифрование по RSA, а также записывается в собственный «лог» файл.

1. kp.sh

В начале инициализируем необходимые таблицы базы данных. Далее следует основной цикл станции. В нем КП считывает зашифрованные файлы из директориях каждой системы, расшифровывает и записывает информацию в базу и в общий лог файл. Если к КП перестали приходить сообщения о работоспособности какой-то станции, то он записывает в лог сообщение, что станция прекратила работу.

5. manage.sh

В данном скрипте происходит создание необходимых временных директорий и запуск всех станций системы, а также остановка всех станций системы и удаление временных директорий.

# 5 Топографическая карта

На рисунке 1 показана топографическая карта с нанесёнными зонами и параметрами работы комплекса ВКО.



Рисунок 1- топографическая карта

# 

# 6 Отрывки журналов работы

# 

# Вывод

В ходе выполнения данной курсовой работы были применены навыки по написанию скриптов на языке Bash, а также была разработана и протестирована система ВКО.