# Плотников Артём 9/2-РПО-23/1

## 1. Переходы

Переходы - это инструкции, изменяющие стандартное последовательное выполнение программы. Они позволяют перескакивать на другую часть программы, что используется в различных структурах управления, таких как циклы и условные операторы.

### Ветвления в алгоритмах и программах

Ветвление - это структура управления, которая выбирает один из нескольких путей в зависимости от заданного условия. Ветвление бывает двух типов:
- **Условные переходы:** выполняются только при выполнении определенного условия.
- **Безусловные переходы:** выполняются независимо от условий и всегда направляют выполнение программы к указанному месту.

### Безусловные переходы

Безусловные переходы - это инструкции, которые просто меняют адрес следующей исполняемой команды на указанный в них. Они не зависят от какого-либо условия. Примеры: команды GOTO в языке BASIC, JMP в ассемблере, которые просто перескакивают на определенную строку или адрес программы.

### Условные переходы. Критерии результата

Условные переходы проверяют некоторое условие и в зависимости от его истинности переходят к другой части кода. Основные критерии результата:
- Результат сравнения (например, a > b, `x == 0`).
- Результат проверки логического выражения (`true/false`).

if в большинстве языков программирования.

### Переходы по смещению: безусловные и условные

Переходы по смещению выполняются относительно текущего адреса, на основе заранее определенного смещения. Это позволяет гибко работать с позициями перехода.
- **Безусловный переход по смещению:** аналогично обычному безусловному, но смещение позволяет экономить ресурсы.
- **Условный переход по смещению:** переход по смещению выполняется только при выполнении условия, что обеспечивает большую гибкость и компактность.

## 2. Циклы

Циклы - это конструкции, позволяющие многократно выполнять один и тот же блок кода. Основные виды циклов:

### Виды циклов

- **Цикл с фиксированным количеством повторений:** заранее известно, сколько раз должен выполниться цикл. Пример: for в большинстве языков.
- **Итерационный цикл:** выполнение цикла продолжается до тех пор, пока выполняется определенное условие. Пример: while и do while.
- **Цикл смешанного типа:** сочетает фиксированное количество повторений и условие окончания. Такой цикл можно встретить в сложных алгоритмах.
- **Мультипликативный цикл:** число повторений увеличивается или уменьшается по мультипликативному принципу. Применяется в алгоритмах, где требуется геометрическая прогрессия, например, в алгоритмах с экспоненциальным ростом количества итераций.

## Цикл с фиксированным количеством повторений

Чаще всего реализуется конструкцией for. Используется для выполнения определенного числа шагов, заданного заранее.

```
for i in range(10):
print(i)
```

## Итерационный цикл

Итерационный цикл продолжается до тех пор, пока соблюдается заданное условие.

```
while condition:
# действия
```

## Цикл смешанного типа

Смешанный цикл можно реализовать с использованием счетчика и условия, например, сочетая for и if.

## Мультипликативный цикл

Этот цикл выполняется с изменяющимся шагом, где количество итераций изменяется в геометрической прогрессии.

```
n = 1
while n < 1000:
print(n)
n *= 2
```

# 3. Переадресация

Переадресация в программировании - это изменение адреса, на который ссылается переменная. Используется для экономии памяти и ускорения доступа к данным.

## Переадресация с использованием констант и восстановление

Константная переадресация указывает на фиксированный адрес в памяти, где хранится необходимое значение. Восстановление происходит, когда переменная или значение возвращаются к исходному адресу.

## Косвенная адресация

Косвенная адресация используется для ссылок на адреса памяти через другую переменную или регистр. Ну для динамической работы с памятью, особенно в циклах.

## Автоинкремент и автодекремент

- **Автоинкремент** - увеличение адреса на определенную величину после каждого обращения.
- **Автодекремент** - уменьшение адреса.

## Стек

Стек - структура данных, которая работает по принципу «последний вошел - первый вышел» (LIFO). Он используется для хранения временных данных и возвращения к ним позже, например, при рекурсивных вызовах.

### Индексный регистр

Индексный регистр - специальный регистр процессора, который хранит смещение или индекс для косвенной адресации, что упрощает работу с массивами.

## 4. Продвинутые команды управления циклом

Эти команды комбинируют управление и циклы для оптимизации программных конструкций.

### Комбинированные команды: управление и индексирование

Эти команды позволяют комбинировать инструкции управления и изменение индекса, что ускоряет выполнение циклов. Пример: выполнение for с шагом, заданным индексом.

### Управление с инкрементом индекса

Команды, которые изменяют значение индекса после каждой итерации.

```
for i in range(0, 10, 2): # шаг 2
print(i)
```

### Управление с использованием счетчика

Счетчик используется для отслеживания числа итераций. Он может быть внутренним элементом цикла.

```
counter = 0
while counter < 10:
print(counter)
counter += 1
```

### Управление с индексированием и счетчиком

Это комбинация, в которой используется как индекс, так и счетчик. Применяется, когда требуется более сложное управление итерациями.

```
index, counter = 0, 0
while counter < 10:
print(index)
index += 2 # увеличиваем индекс на 2
counter += 1
```