Задание 4

Реализовать функцию печати условного IP-адреса.

Условность его заключается в том, что количество элементов не обязательно должно быть равно 4-ём или 8-ми, а также каждый элемент не обязательно должен быть числом из диапазона 0..255. От идеи IP-адреса остаётся фактически только вывод элементов через `.` (символ точки).

Функцию нужно реализовать для различных входных параметров путём использования механизма SFINAE. Всего нужно выполнить 3 обязательных и один опциональный вариант функции.

1. Адрес может быть представлен в виде произвольного целочисленного типа. Выводить побайтово в беззнаковом виде, начиная со старшего байта, с символом `.` (символ точки) в качестве разделителя. Выводятся все байты числа.
2. Адрес может быть представлен в виде строки. Выводится как есть, вне зависимости от содержимого.
3. Адрес может быть представлен в виде контейнеров `std::list`, `std::vector`. Выводится полное содержимое контейнера поэлементно и разделяется `.` (символом точка). Элементы выводятся как есть.
4. Опционально адрес может быть представлен в виде `std::tuple` при условии, что все типы одинаковы. Выводится полное содержимое поэлементно и разделяется `.` (одним символом точка). Элементы выводятся как есть. В случае, если типы кортежа не одинаковы, должна быть выдана ошибка при компиляции кода.

Прикладной код должен содержать следующие вызовы:

```
print_ip( int8_t{-1} ); // 255
print_ip( int16_t{0} ); // 0.0
print_ip( int32_t{2130706433} ); // 127.0.0.1
print_ip( int64_t{8875824491850138409} );// 123.45.67.89.101.112.131.41
print_ip( std::string{“Hello, World!”} ); // Hello, World!
print_ip( std::vector<int>{100, 200, 300, 400} ); // 100.200.300.400
print_ip( std::list<shot>{400, 300, 200, 100} ); // 400.300.200.100
print_ip( std::make_tuple(123, 456, 789, 0) ); // 123.456.789.0
```

Дополнительное задание

Добавить в пайплайн сборки на Github Actions вызов `doxygen` и публикацию html-версии документации на github-pages. Пара инструкций по тому, как это можно сделать:

https://ntamonsec.blogspot.com/2020/06/github-actions-doxygen-documentation.html
https://dev.to/denvercoder1/using-github-actions-to-publish-doxygen-docs-to-github-pages-177g
https://wiki.jlab.org/epsciwiki/images/a/aa/HostingDocsOnGithub.pdf

Включить в репозиторий файл `Doxyfile` с включенными опциями `HAVE_DOT` и `EXTRACT_ALL`.

Дополнительные требования

- функция печати должна быть одной шаблонной функцией, разные варианты входов должны быть реализованы через механизм SFINAE
- вариант для целочисленного представления должен представлять собой одну функцию
- вариант для контейнеров std::list и std::vector должен представлять собой одну функцию
- не должно быть реализации типа "если не совпало с иными - значит это контейнер"
- найдите самый простой способ для печати `std::string` (но функция всё ещё должна быть шаблонной)
- опциональная реализация для `std::tuple` должна приводить к ошибке в случае отличающихся типов
- не должно быть ограничений на размер целочисленных типов (в байтах), на размер контейнеров и кортежа (количество элементов)
- бинарный файл и пакет должны называться `print_ip`

Проверка

Задание считается выполненным успешно, если после анализа кода на соответствие требованиям о реализации, установки пакета и запуска приложения на экране будет сформирован вывод (в соответствии с требованиями к прикладному коду):

```
255
0.0
127.0.0.1
123.45.67.89.101.112.131.41
Hello, World!
100.200.300.400
400.300.200.100
123.456.789.0
```

В случае выполнения дополнительного задания должна быть опубликована сгенерированная doxygen-ом документация. После успешной публикации документация становится доступной по адресу:

https://***username***.github.io