Московский авиационный институт

(Национальный исследовательский университет)

**Лабораторные работы 1 - 10**

**Архитектура программной среды**

**Лаборатория 1.** Main функция, Абстракция функции, Черный ящик и блок, Основные типы: int, float

**Лаборатория 2.** Абстракция функции, вложенные блоки, комментарии, Создание переменных (объявление переменных) Создание функции (3 шага), Назначение результата переменной, результат печати. Вызов функции и результат печати. Глобальные и локальные переменные (например, вложенные блоки). Типы.

**Лаборатория З.** Инициализация переменных. Арифметические операторы. Преобразование типов. Операторы инкремента/дикремента. Расширенные операторы. Приоритет & Ассоциативность операторов. Диапазон типов данных. Автоматическое преобразование типов. Явное преобразование. Базовый ввод scanf\_s.

**Лаборатория 4.** Логические операторы И, ИЛИ, НЕТ -!. Булева алгебра. Логические операторы Приоритет. Примеры.

**Лаборатория 5.** Блок схемы: пример ввода и вывода. Таблица реляционных операторов. Циклы: for (выражение инициализации; условие; инкремент), диаграммы цикла, инструкции for с несколькими инициализируемыми переменными,; оператор do цршду; while (управляющее выражение); вложенные циклы, оператор continue и break.

**Лаборатория 6.** Пинятие решений: if оператор ,if-else оператор , if-else if else. Лестница if-else if else, switch (..) оператор, тернарный оператор.

**Laboratory 7.** Переменные массива, инициализация массива, статический массив, многомерные массивы, инициализация многомерных массивов.

**Лаборатория 8.** 3d\_array. Массивы символов. Инициализация строк при объявлении, scanf, по strcpy. Строка функций I/O (gets\_s(фраза), присвоить значение переменной(фраза).

**Laboratory 9.** Функции I/O символов. Библиотечные функции "Math". Функции определяемые пользователем. Рекурсия. Форматированный ввод и вывод. Перечислимые типы и переменные.

**Laboratory 10.** Переменные типа указатель. Операторы взятие значения переменной по адресу. Арифметика указателей. "Вызов-по-Ссылке и вызов-по-значению" аргументов. Указатели и массивы. Массивы как аргументы функций.

**\*Laboratory 11.** Структураstructure (9.1), Класс сlass (9.2), Перегрузка Overloading (9.3)

**\*Laboratory 12.** Объединение union, Аллокация памяти, Командная строка, Ввод - вывод в файл .(l0.1-4 )

\_\_\_\_\_\_\_\_/Калугин К.А. / Подпись: \_\_\_\_\_\_\_\_/Семенов А.С. / Подпись: Дата : \_\_ 2024

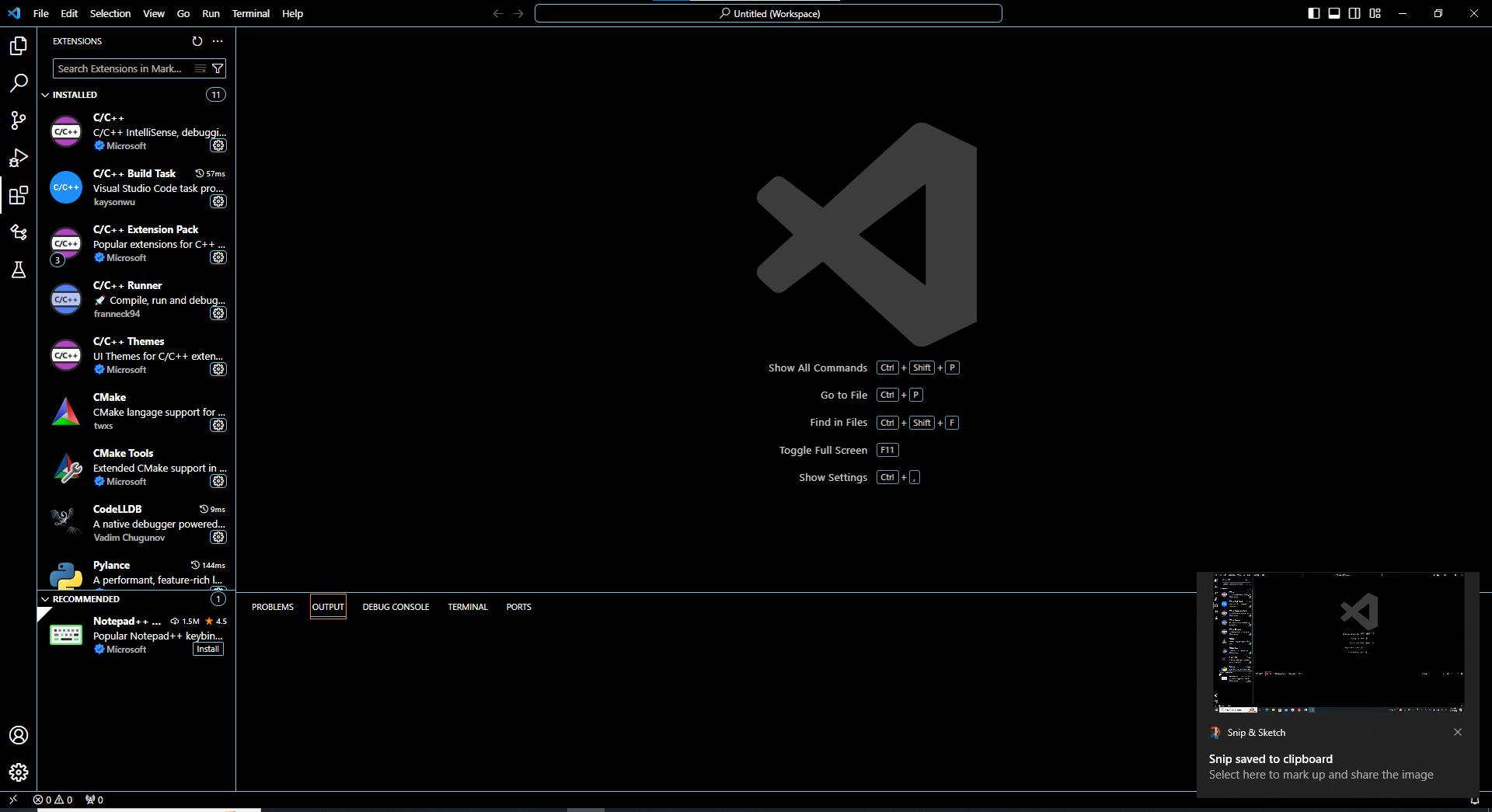
**Лаборатория 1.**  main функция, Абстракция функции, Черный ящик и блок, Основные типы: int, float

**Visual Studio Code (VSC)** — это бесплатный и открытый кроссплатформенный редактор кода, разработанный Microsoft для Windows, Linux и macOS. Он включает в себя множество функций, которые делают его популярным выбором для разработчиков, в том числе:

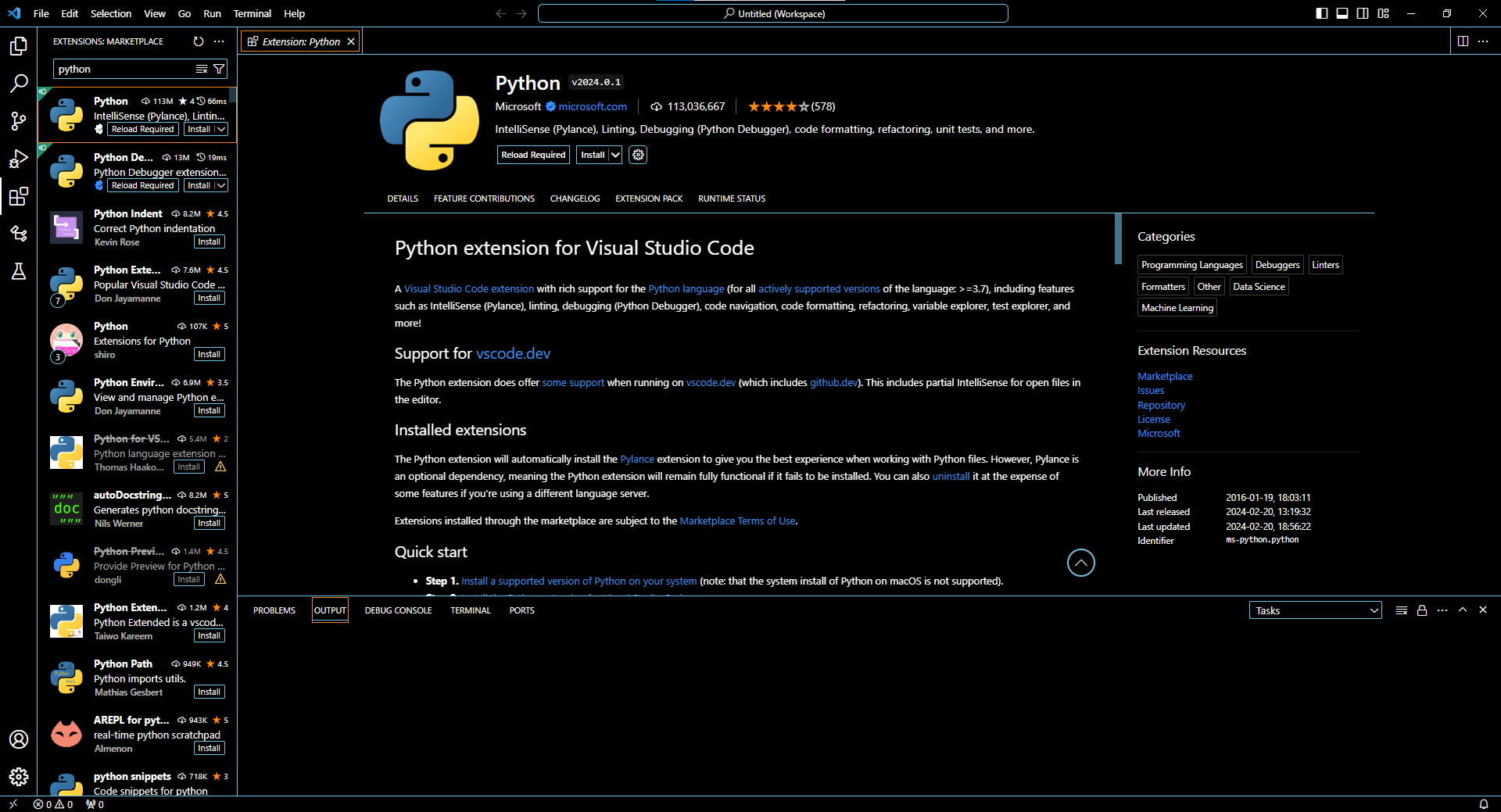
* **Поддержка нескольких языков:** VSC поддерживает широкий спектр языков программирования, включая JavaScript, Python, Java, C++, C# и другие.
* **Интеллектуальное автозаполнение:** VSC предоставляет интеллектуальное автозаполнение кода, которое помогает разработчикам писать код быстрее и точнее.
* **Отладка:** VSC включает в себя мощные инструменты отладки, которые позволяют разработчикам легко находить и исправлять ошибки в своем коде.
* **Интеграция с Git:** VSC тесно интегрирован с Git, что позволяет разработчикам управлять своим кодом и сотрудничать с другими.
* **Расширяемость:** VSC можно расширить с помощью широкого спектра расширений, которые добавляют новые функции и возможности.
* **Легкий и быстрый:** VSC является легким и быстрым редактором кода, что делает его идеальным для работы с большими и сложными проектами.
* **Открытый исходный код:** VSC — это проект с открытым исходным кодом, что позволяет разработчикам вносить свой вклад и создавать собственные расширения.

Для создания нового Python проекта в Visual Studio Code необходимо выполнить следующие действия:

1. **Установите Python.** Для этого необходимо нажать комбинацию клавиш Ctrl+Shift+X или нажать соответствующую кнопку в интерфейсе программы. Откроется следующее окно:



В списке слева необходимо найти расширение под названием «Python» и установить его при помощи кнопки «Install».



После этого необходимо выполнить перезагрузку VSC.

1. **Создайте новую папку для проекта:** Откройте VSC и создайте новую папку для вашего Python-проекта.
2. **Откройте папку в VSC:** Нажмите на меню "Файл" > "Открыть папку" и выберите папку вашего проекта.
3. **Создайте виртуальное окружение (необязательно):** Виртуальное окружение изолирует зависимости проекта от системных зависимостей. Чтобы создать виртуальное окружение, откройте терминал в VSC и выполните следующую команду:

python -m venv venv

1. **Активируйте виртуальное окружение (необязательно):** Чтобы активировать виртуальное окружение, выполните следующую команду в терминале:

source venv/bin/activate

1. **Создайте файл Python:** Щелкните правой кнопкой мыши в папке проекта и выберите "Создать" > "Файл Python". Назовите файл main.py.
2. **Напишите код Python:** В файле main.py напишите следующий код Python:

print("Hello, world!")

1. **Запустите код Python:** Нажмите клавишу F5 или перейдите в меню "Выполнить" > "Запустить без отладки". VSC запустит код Python и выведет "Hello, world!" в терминале.

Теперь у вас есть Python проект в VSC. Для его хранения можно использовать систему GitHub.

**GitHub** — это веб-сервис для хостинга и совместной разработки программного обеспечения с использованием контроля версий Git. Он предоставляет разработчикам централизованное место для хранения своего кода, отслеживания изменений и совместной работы над проектами.

**Ключевые термины:**

* **Репозиторий:** Хранилище кода на GitHub.
* **Запрос на вытягивание (Pull Request):** Предложение по внесению изменений в репозиторий.
* **Ветка:** Копия репозитория, в которой можно вносить изменения, не влияя на основную ветку.
* **Слияние:** Объединение изменений из ветки в основную ветку.

**Как создать репозиторий на GitHub:**

1. **Создайте учетную запись GitHub:** Если у вас еще нет учетной записи GitHub, перейдите на сайт GitHub и создайте ее.
2. **Откройте GitHub:** Перейдите на сайт GitHub и войдите в свою учетную запись.
3. **Нажмите кнопку "Новый репозиторий":** В правом верхнем углу главной страницы GitHub нажмите кнопку "Новый репозиторий".
4. **Введите имя репозитория:** Введите желаемое имя для своего репозитория.
5. **Добавьте описание (необязательно):** Вы можете добавить описание своего репозитория, чтобы дать другим разработчикам представление о его назначении.
6. **Выберите параметры инициализации (необязательно):** GitHub предлагает различные параметры инициализации, такие как добавление файла README, лицензии или .gitignore. Вы можете выбрать нужные вам параметры.
7. **Нажмите кнопку "Создать репозиторий":** После того, как вы заполнили все необходимые поля, нажмите кнопку "Создать репозиторий".

**Лаборатория 2.** Абстракция функции, вложенные блоки, комментарии, Создание переменных (объявление переменных) Создание функции (3 шага), Назначение результата переменной, результат печати. Вызов функции и результат печати. Глобальные и локальные переменные (например, вложенные блоки). Типы.

