Введение. Часть 2

Переменные и функции

Сулыз Андрей 3-4Б
iistclub.ru

<Переменные>

*html тег имеет вид типа <> </>. Тег открывается и закрывается соответственно.

Прелюдие

Прелюдие

Тип: Исторические данные

Назовем этот ящик: “*Данные за 1998 год*”

В архиве этот ящик находится по адресу: 1 этаж, 2-й стеллаж, 3 полка, 6 ряд

Тип: Данные по химии

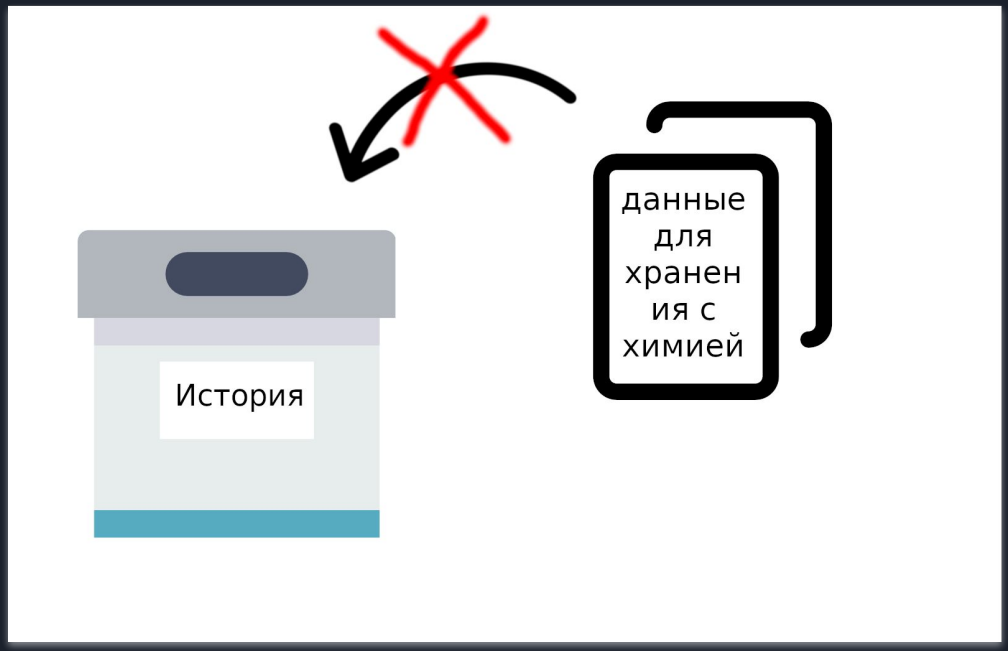
Назовем этот ящик: “*Достижения в хим промышленности в ЮРГПУ 1990*”

В архиве этот ящик находится по адресу: 3 этаж, 12-й стеллаж, 1 полка, 9 ряд

Прелюдие

После того как ящик создан и ему установлен определенный тип (история, химия и т.д.) нам **НЕЛЬЗЯ** изменить его данные на другой тип.

Например: Имея ящик с типом **ИСТОРИЯ** мы не можем хранить в нем данные по химии, *но **МОЖЕМ** изменить его содержание на другие данные по истории.*



Виды внутренней компьютерной памяти

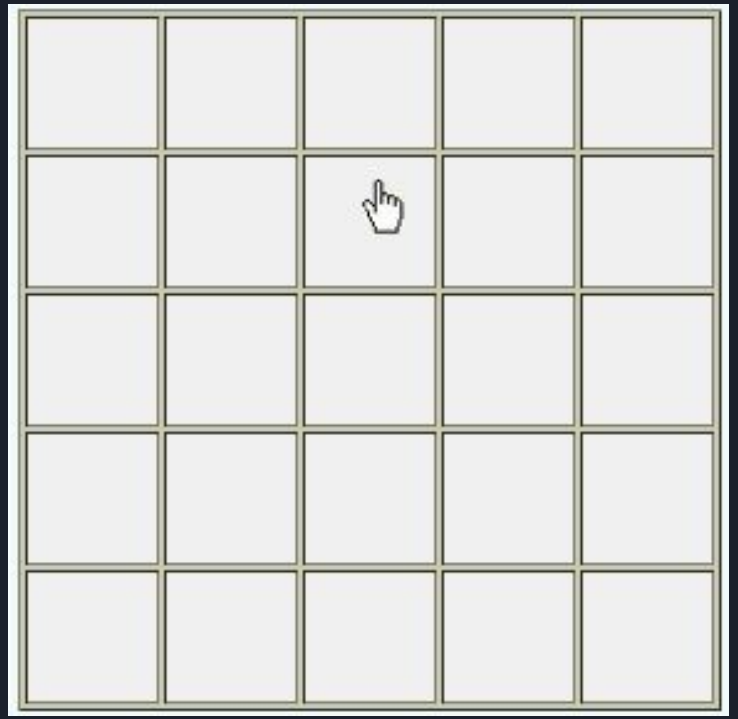
Внутренняя память называется так потому, что она встроена в основные блоки компьютера и является неотъемлемым элементом системы, обеспечивающим ее работоспособность. Удалить или извлечь ее без негативных последствий невозможно. Различают следующие ее виды:

- ***оперативная***
- ***кэш-память***
- ***постоянная***
- ***полупостоянная***
- ***видеопамять***

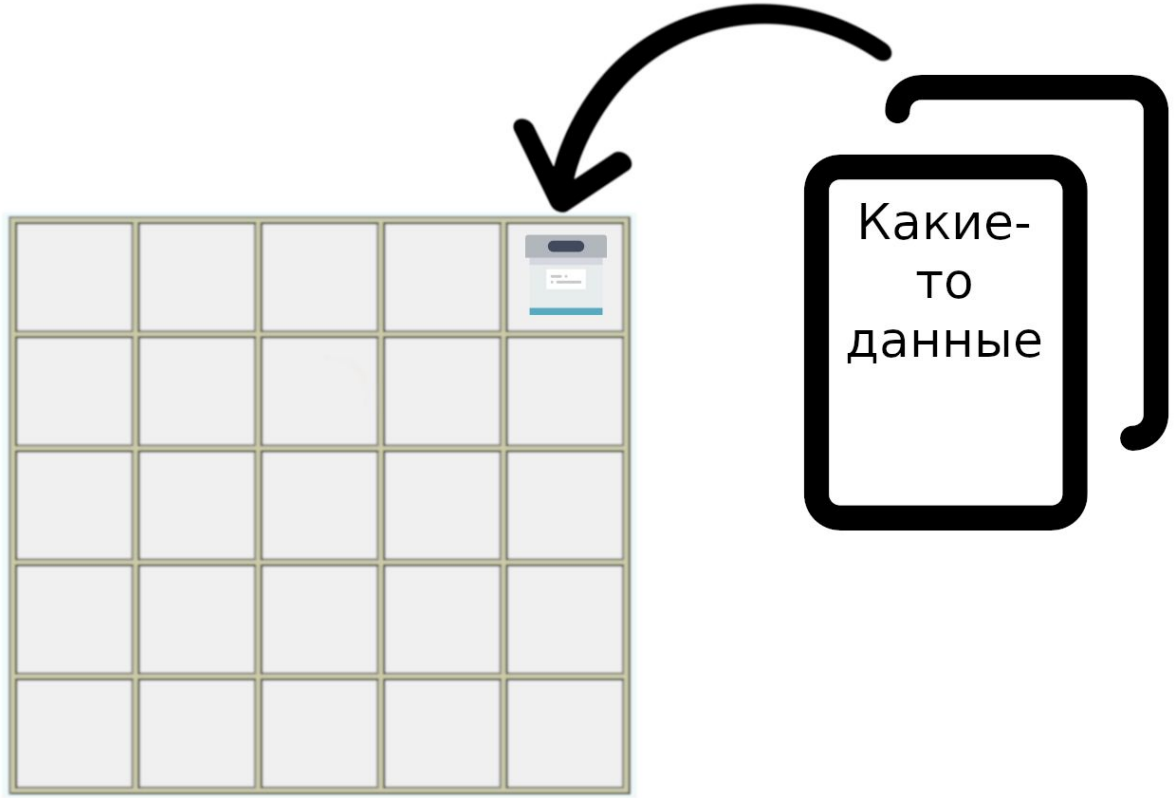
Источник: http://kak-bog.ru/vidy-pamyati-kompyutera

Структура оперативной памяти (ОЗУ, RAM)

Ядро микросхемы динамической памяти состоит из множества ячеек, каждая из которых хранит всего один бит информации. На физическом уровне ячейки объединяются в прямоугольную **матрицу**, горизонтальные линейки которой называются строками (**ROW**), а вертикальные - столбцами (**Column**) или страницами (**Page**).



Источник: http://citforum.ru/book/optimize/ram.shtml


Какие-
то
данные

Переменные

Переме́нная — поименованная, либо адресуемая иным способом область памяти, адрес которой можно использовать для осуществления доступа к данным. Данные, находящиеся в переменной (то есть по данному адресу памяти), называются значением этой переменной.

Переменные нужны для того, чтобы они хранили в себе некоторое значение, которое может изменяться со временем.

Пример использования переменных

```
1  print(2)
2  print(2 + 3)
3  print(2 / 123)
4  print(2 * 12)
5  print(2 + ' Cool!')
6  print(2 + 2 - 2 + 2)
```

Пример без использования переменных

```
1  number = 2
2
3  print(number)
4  print(number + 3)
5  print(number / 123)
6  print(number * 12)
7  print(number + ' Cool!')
8  print(number + number - number + number)
```

Пример с использованием переменных

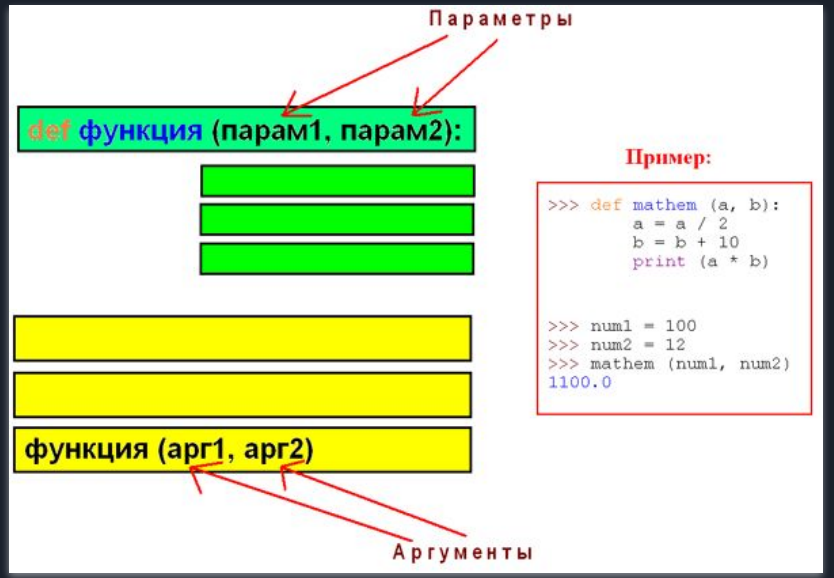
</Переменные>

<Функции>

Функции

Фу́нкция в программировании — фрагмент программного кода (**подпрограмма**), к которому можно обратиться из другого места программы.

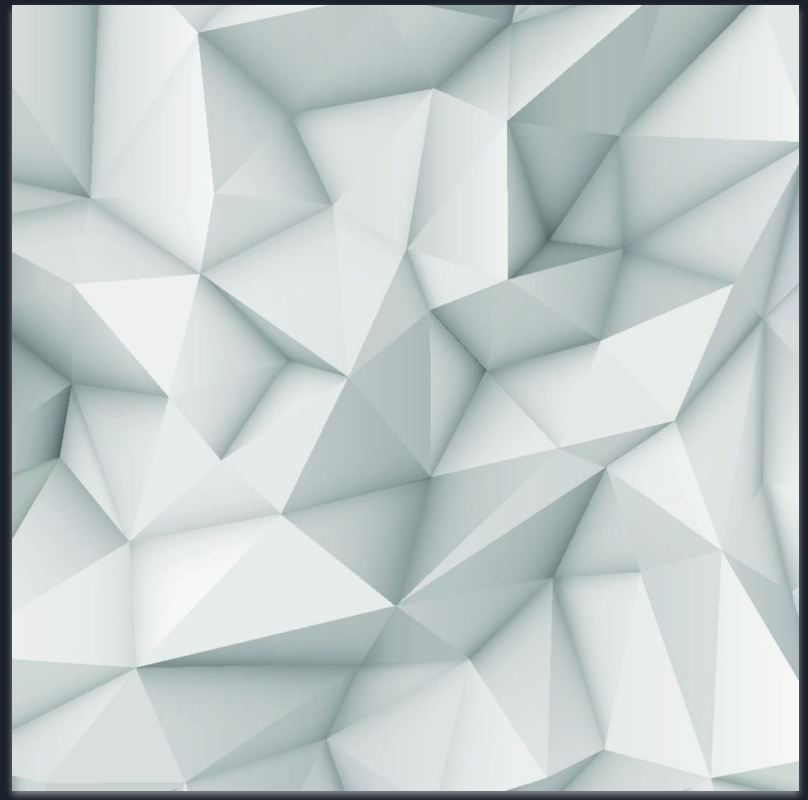
Функция может принимать параметры и должна возвращать некоторое значение, возможно пустое. Функции, которые возвращают пустое значение, часто называют процедурами. В некоторых языках программирования объявления функций и процедур имеют различный синтаксис, в частности, могут использоваться различные ключевые слова.



Абстракция

Абстра́кция (лат. abstractio — отвлечение) — теоретическое обобщение как результат абстрагирования.

Абстрагирование — отвлечение в процессе познания от несущественных сторон, свойств, связей объекта (предмета или явления) с целью выделения их существенных, закономерных признаков. Результат абстрагирования — абстрактные понятия, например: цвет, кривизна, красота и т. д.



Объявление функции

```
11  def add_two_numbers(num1, num2):
12      result = num1 + num2
13      print(result)
```

Объявление функции осуществляется с помощью ключевого слова **def**, после которого следует название функции (**оно может быть любым!**), затем в круглых скобках определяются **параметры функции**, после чего следует знак **“:”** и ниже указывается сам код функции.

Вызов функции

```
11  def add_two_numbers(num1, num2):
12      result = num1 + num2
13      return result
14
15  result_adding = add_two_numbers(2, 10)
16  print(result_adding)
```

Вызов функции осуществляется с помощью указания имени функции и затем в круглых скобках указания аргументов функции.

Обычно из функции возвращается какое-то значение, результат работы функции. Его можно записать в новую переменную

Избегание использования функций

```python
def add_two_numbers(num1, num2):
    result = num1 + num2
    result = result * num2
    result = result / num1

    return result

result_adding1 = add_two_numbers(2, 10)
result_adding2 = add_two_numbers(5, 19)
result_adding3 = add_two_numbers(80, 20)
```

Код с использованием функции

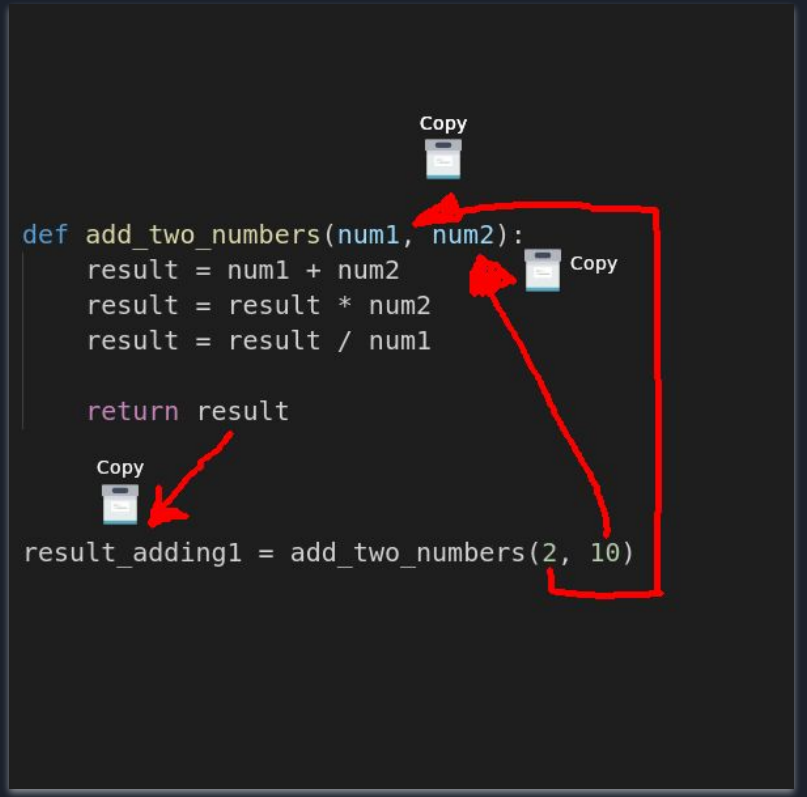
```python
result_adding1 = 2 + 10
result_adding1 = result_adding1 * 10
result_adding1 = result_adding1 / 2

result_adding2 = 5 + 19
result_adding2 = result_adding2 * 19
result_adding2 = result_adding2 / 5

result_adding3 = 80 + 20
result_adding3 = result_adding3 * 20
result_adding3 = result_adding3 / 80
```

Тот же код, но без использования функции

Как работает передача значений в функциях

В качестве параметров в функции приходят **КОПИИ** переменных переданных при вызове данной функции.

Из функции возвращается значение с помощью ключевого слова **return** и далее указывается переменная. Возвращается также копия переменной используемой в функции.

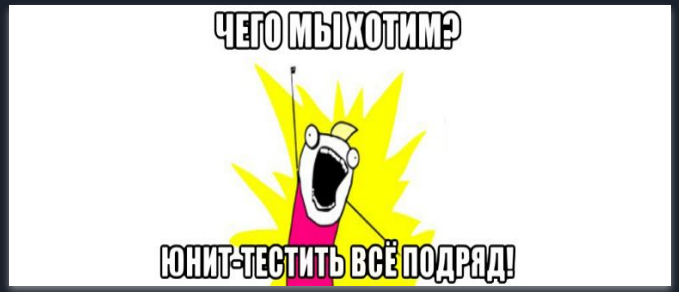


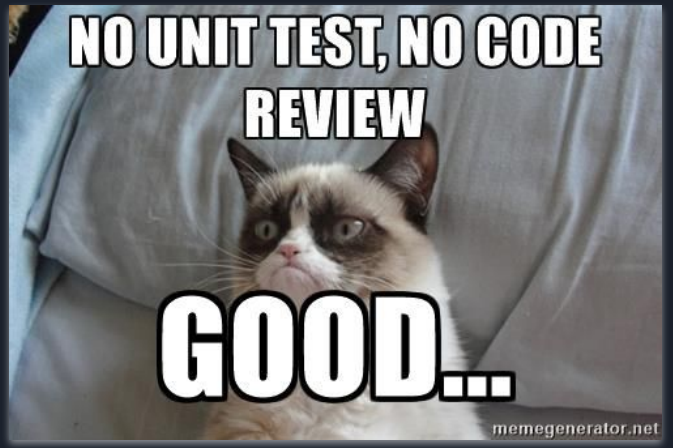
Юнит-тестирование

Автоматические тесты дают уверенность, что ваша программа работает как задумано. Такие тесты можно запускать многократно. Успешное выполнение тестов покажет разработчику, что его изменения не сломали ничего, что ломать не планировалось.

Провалившийся тест позволит обнаружить, что в коде сделаны изменения, которые меняют или ломают его поведение. Исследование ошибки, которую выдает провалившийся тест, и сравнение ожидаемого результата с полученным даст возможность понять, где возникла ошибка, будь она в коде или в требованиях.

TDD - разработка через тестирование





</Функции>

Дополнительно

AR. Дополненная реальность.