Динамическая память, примитивы работы с ней и принцип функционирования.

Память выделяется: статическая – во время загрузки программы, автоматическая – когда поток управления программы входит в блок. Язык Си позволяет работать с выделенной памятью.

```
void* malloc (size_t size)
```

***size_t* size** – значение в байтах, выделяемого блока памяти.
Возвращаемое значение, указатель на начало блока.
Функция **malloc()** – ищет подходящий по размеру блок свободной памяти. Память будет анонимной, т.к. **malloc()** не назначает выделенному блоку имя.
Следовательно, требуется результат (**void***) функции **malloc()** присвоить переменной типа указателя для доступа к этой памяти.

void* – обобщенный указатель, тип которого должен быть изменен в соответствии с объектом данных размещаемом в выделенной памяти.

Благодаря этому свойству, функция malloc может применятся для выделение памяти под массивы и структуры и т.п. Ведь указатель можно привести к подходящему типу.

```
double *arr = NULL;
arr = (double *)malloc(10 * sizeof(double));
```

При использовании **malloc()** требуется явным образом выполнять приведение типов, но присваивание значения указателя на ***void**** указателю другого типа не является конфликтом типов.
Если не удается найти запрошенный блок памяти, **malloc()** возвращает нулевой указатель.

```
double *arr = NULL;
arr = (double *)malloc(10 * sizeof(double));
if (arr == NULL)
{
    puts("Не удалось выделить память.")
    exit(EXIT_FAILURE);
}
```

Каждый вызов функции **malloc()** должен сопровождаться вызовом процедуры **free()**. В качестве единственного аргумента принимает адрес, возвращенный ранее функцией **malloc()**. Вызов процедуры **free()** освобождает память, которая была выделена.

```
double *arr = NULL;
arr = (double *)malloc(10 * sizeof(double));
free(arr);
```

Прототипы **malloc()** и **free()**, находятся в заголовочном файле **stdlib.h**.

Функция **calloc()**, предлагает ещё один способ выделения памяти.

```
void* calloc(size_t num, size_t size);
```

где ***num*** – кол-во элементов, а ***size*** размер элемента в байтах. Особенность **calloc()** заключается в том, что блок инициализирует все биты нулями.

```
int *pData, i;
scanf(%d, &i);
pData = (int *)calloc(i, sizeof(int));
if (pData == NULL)
    exit(EXIT_FAILURE);
```

Функция **realloc()**

```
void* realloc(void* ptr, size_t new_size);
```

- Изменяет размер ранее выделенного блока памяти.
- Может переместить данные в новый блок, если требуется.

```
arr = realloc(arr, 20 * sizeof(int)); // Увеличение массива до 20 элементов
```

Частые ошибки:

- Утечка памяти (не очищена память после ее использования)
  Решение: прописать **free()** после окончания работы с динамической памятью