1. Введение

В ходе комплексного тестирования программных средств возникает необходимость интерпретации результатов множества тестов, написанных по различным правилам и для различных целей. Для решения задачи автоматизации запуска, сбора информации и интерпретации результатов тестирования необходимо привести интерфейсную часть всех тестирующих программ к единообразному виду позволяющему с наименьшими затратами решать поставленную задачу. Для данных целей предлагается использовать единую библиотеку с небольшим прикладным программным интерфейсом (API), исключающую возможность административного взаимонепонимания при реализации правил для обработки входных параметров и систематизации выходных данных в средствах тестирования и диагностики. При этом, в силу того что приложения тестирования могут взаимодействовать с оборудованием, необходимо обеспечить полную поддержку многопоточности в такой библиотеке.

1. Специальная часть
   1. Анализ исходных требований для разрабатываемой библиотеки обработки входных параметров и систематизации выходных данных
   2. Разработка соглашений о вызовах функций библиотеки
      1. Разработка соглашений о вызовах функций обработки ошибок работы библиотеки
      2. Разработка соглашений о вызовах функций получения входных параметров программ тестирования

Для того чтобы автоматизировать получение параметров командной строки предлагается использовать семейство функций tioGet\*иtioGetDef\*, где вместо знака «\*» должна быть подставлена одна из следующих букв, означающих какого типа будет возвращаемое значение:

* L – long
* D – double
* C – char
* S – char\* (string)

Коды ошибок в результате работы функций содержатся в таблице 2.1, приведенной ниже.

|  |  |
| --- | --- |
| TENOPAR | Параметр не зарегистрирован при инициализации библиотеки |
| TEINCTYPE | Параметр не может быть приведен к запрошенному типу |
| TENOTSET | Параметр не передан при вызове приложения. |
| TENES | Размер буфера недостаточно велик для помещения параметра |
| TEFAILL | Отказ по непонятным причинам |

**Функции tioGetS и tioGetDefS**

|  |  |  |
| --- | --- | --- |
| int tioGetS | ( | const char\* name, |
|  |  | char\* buff, |
|  |  | size\_t buff\_len |
|  | ) |  |

Функция получения параметра командной строки в форме последовательности символов. name – указатель на имя параметра, значение которого необходимо получить. buff – указатель на адрес памяти, куда функция поместит значение искомого параметра в виде последовательности символов. buff\_len – переменная, содержащая значение максимальной длины строки.

Возвращает 0 в случае успешного выполнения. В противном случае возвращаемое значение примет вид кода ошибки из таблицы 2.1. При возникновении любой из ошибок функция tioGetS заносит в buff нулевой символ.

|  |  |  |
| --- | --- | --- |
| int tioGetDefS | ( | const char\* name, |
|  |  | const char\* default, |
|  |  | char\* buff, |
|  |  | size\_t buff\_len |
|  | ) |  |

Функция получения параметра командной строки в форме последовательности символов. name – указатель на имя параметра, значение которого необходимо получить. default – значение параметра, связанного с именем name по умолчанию. buff – указатель на адрес памяти, куда функция поместит значение искомого параметра в виде последовательности символов. buff\_len – переменная, содержащая значение максимальной длины строки.

В случае если значение, связанное с именем namе получить не удалось, то в буфер buff присваивается значение параметра default.

Возвращает 0 в случае успешного выполнения. В противном случае возвращаемое значение примет вид кода ошибки из таблицы 2.1. При возникновении любой из ошибок функция tioGetDefS заносит в buff нулевой символ.

**Функции tioGetL и tioGetDefL**

long tioGetL ( const char\* name )

Функция возвращает значение параметра командной строки, связанного с именем name. Значение должно быть расположено в промежутке от минимально допустимого для типа long до предшествующего максимально допустимому значению для типа long (от LONG\_MIN до LONG\_MAX-1). В случае если такого параметра нет, или значения параметра не находятся в указанном промежутке, или не могут быть приведены к типу данных long, возвращается максимально допустимое значение для типа long. Код ошибки в этом случае может быть получен с помощью функции tioGetError().

Возможные ошибки: TENOTSET и TEINCTYPE.

|  |  |  |
| --- | --- | --- |
| long tioGetDefL | ( | const char\* name, |
|  |  | const long default |
|  | ) |  |

Функция возвращает значение параметра командной строки, связанного с именем name. Значение должно быть расположено в промежутке от минимально допустимого для типа long до предшествующего максимально допустимому значению для типа long (от LONG\_MIN до LONG\_MAX-1). В случае если такого параметра нет, или значения параметра не находятся в указанном промежутке, или не могут быть приведены к типу данных long, возвращается значение по умолчанию присвоенное при вызове функции параметру default. Код ошибки в этом случае может быть получен с помощью функции tioGetError().

Возможные ошибки: TENOTSET и TEINCTYPE.

* + 1. Разработка соглашений о вызовах функций обработки выходных данных программ тестирования
    2. Проектирование методов аварийного завершения программ, использующих разрабатываемую библиотеку
  1. Реализация функций разрабатываемой библиотеки
  2. Прототипирование среды исполнения подпрограмм библиотеки

1. Технологическая часть
   1. Профилирование разрабатываемого программного обеспечения
   2. Анализ производительности библиотеки интерфейсов
   3. Отладка и тестирование разрабатываемой библиотеки
2. Охрана труда и окружающей среды. Разработка мероприятий по обеспечению благоприятных санитарно-гигиенических условий труда инженера
3. Экономическая часть. Обоснование экономической эффективности разработки библиотеки функций унификации процессов обработки входных параметров и систематизации выходных данных
4. Заключение