ЛАБОРАТОРНАЯ РАБОТА 6

ПРОСТЕЙШИЙ ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

1. Цель.

Целью лабораторной работы является изучение функций muLISP'a (newLISP-tk), обеспечивающих создание пользовательского интерфейса.

2. Краткие теоретические сведения.

2.1 Работа с экраном дисплея в muLISP'e.

Для работы с экраном дисплея в muLISP'e существует ряд встроенных функций [1], которые обеспечивают управление выводом на экран в текстовом режиме MS DOS. Рассмотрим важнейшие из них.

Функция (CLEAR-SCREEN) очищает текущее окно, перемещает курсор в верхний левый угол окна и возвращает T.

Функция (SET-CURSOR) возвращает список из двух целых чисел, задающих позицию курсора.

Функция (SET-CURSOR ROW COLUMN) перемещает курсор в строку с номером ROW и колонку с номером COLUMN. Если 0<=ROW<25 и 0<=COLUMN<80, то функция перемещает курсор в требуемую строку и колонку и выдает T. В противном случае возвращается NIL.

Функции (ROW) и (COLUMN) возвращают, соответственно, номер строки и номер колонки, в которых находится курсор относительно текущего окна.

Функция (INSERT-LINES N) в текущем окне вставляет N пустых строк, начиная со строки, помеченной курсором, и возвращает T, если N>=0. В противном случае возвращается NIL.

Функция (DELETE-LINES N) в текущем окне уничтожает N строк, начиная со строки, помеченной курсором, и возвращает T, если N>=0. В противном случае возвращается NIL.

Функция (MAKE-WINDOW ROW COLUMN ROWS COLUMNS) создает на экране прямоугольную область как текущее окно, перемещает курсор в левый верхний угол окна и возвращает T. Верхний левый угол окна определяется значениями аргументов **ROW** и **COLUMN**. Аргумент **ROW** должен быть нулем или положительным целым числом меньшим, чем 25. Аргумент **COLUMN** должен быть нулем или положительным целым числом меньшим, чем 80. И **ROW**, и **COLUMN** по умолчанию принимаются за 0. Аргумент **ROWS** должен быть положительным целым числом меньшим или равным **(25 - ROW)**. **ROWS** по умолчанию рассматриваются как **(25 - ROW)**. Аргумент **COLUMNS** должен быть положительным целым числом меньшим или равным **(80 - COLUMN)**. **COLUMNS** по умолчанию рассматриваются как **(80 - COLUMN)**.

Функция **(MAKE-WINDOW)** возвращает список из четырех элементов: исходной строки, исходной колонки, количества строк и количества колонок текущего окна.

Функция (FOREGROUND-COLOR N) устанавливает цвет выводимых на экран символов (фон переднего плана) в зависимости от значения числа N. Число N соответствует одной из 16 цветовых констант : 0 – черный, 1 – синий, 2 – зеленый и т.д., см. [2], стр.162. Если функции вызываются без аргументов, то они возвращают соответственно код текущего цвета текста.

Аналогичным образом работает функция (BACKGROUND-COLOR N), которая устанавливает цвет бордюра (фон заднего плана).

Функция (DISPLAY-PAGE N) устанавливает в качестве текущей страницу видеопамяти с номером **N** и возвращает в качестве результата номер предыдущей видеостраницы. При этом необходимо, чтобы muLISP работал на компьютере IBM PC с графическим дисплеем, а N<=0<8. В противном случае функция **DISPLAY-PAGE** возвращает номер текущей страницы дисплея. Это дает возможность быстро чередовать содержимое нескольких экранов дисплея, заполненных текстом.

2.2 Работа с экраном дисплея в newLISP-tk.

Для создания окон в newLISP-tk используется функция tk.

Вызов :

(tk "toplevel .mywin")

создает окно верхнего уровня с заголовком «mywin» и размещает его поверх консоли.

Создание окна с масштабной линейкой и печать масштабных коэффициентов :

```
(context 'SCALE)
(define (run )
  (tk "proc ScaleValue { val } { Newlisp \"(silent (SCALE:set-value $val ) )\" }")
  (tk "if {[winfo exists .scaler_for_example] == 1} {destroy .scaler_for_example}")
  (tk "toplevel .scaler_for_example")
  (tk "scale .scaler_for_example.scale -from -10 -to 20 -length 200 -orient horizontal")
  (tk ".scaler_for_example.scale configure -command {ScaleValue}")
  (tk "pack .scaler_for_example.scale"))

(define (set-value x)
 (print x " "))

(context 'MAIN)

(SCALE:run)
```

Создание окна с заголовком «mywin» и фоном зеленого цвета :

```
(tk "set win [toplevel .mywin]; $win config -background green")
```

Вариант :

```
(set 'win (tk "toplevel .mywin"))(tk win " config -background green")
```

Создание окна с кнопкой :

```
(context 'App)

(define (app-example )
  (tk "if {[winfo exists .dw] == 1} {destroy .dw}")
  (tk "toplevel .dw")
  (tk "label .dw.lb -width 20 -height 1")
  (tk "button .dw.quit -text EXIT -command {destroy .dw}")
  (tk "pack .dw.quit -side top -pady 6")
  (tk "bind .dw <Destroy> { wm deiconify . }")
  (tk "wm title .dw { Window with a button !}"))

(context 'MAIN)

(App:app-example)
```

Создание окна с полем для ввода текста :

```
(context 'DemoEntry)

(define (entry )
 (tk "if {[winfo exists .entry] == 1} {destroy .entry}")
 (tk "toplevel .entry")  (tk "wm title .entry {entry example}")
 (tk "entry .entry.e -width 30 -textvariable ::display")
 (tk "bind .entry.e <Return> { Newlisp {(silent (get-entry-text))} }")
 (tk "pack .entry.e"))

(define (get-entry-text )
 (println (tk "set ::display")))

(context 'MAIN)

(DemoEntry:entry)
```

3. Задание на лабораторную работу.

Задание 1.

Написать программу, создающую два окна : окно меню и рабочее окно. Окно меню может быть расположено горизонтально или вертикально, сверху или, соответственно, справа экрана. Размеры экранов выбрать самостоятельно. Меню должно обеспечивать вызовы всех функций, реализованных в лабораторных работах 2 и 3. Ввод исходных данных и вывод результата осуществлять в рабочем окне.

Задание 2.

Вывести на экран прямоугольник, содержащий 6 на 6 квадратов различного цвета. Обеспечить перемещение цветов по квадратам, для нечетных вариантов по горизонтали, четных - по вертикали. Обеспечить движение в обе стороны.

Задание 3.

Вывести на экран прямоугольник, содержащий 6 на 6 квадратов различного цвета. Обеспечить циклическое перемещение цветов по спирали (рис. 1). Из центра должен обеспечиваться переход в начальную позицию. При выполнении задания в среде newLISP-tk за основу рекомендуется взять демо-пример, содержащийся в файле C:\Program Files\newlisp\Demo.lsp.

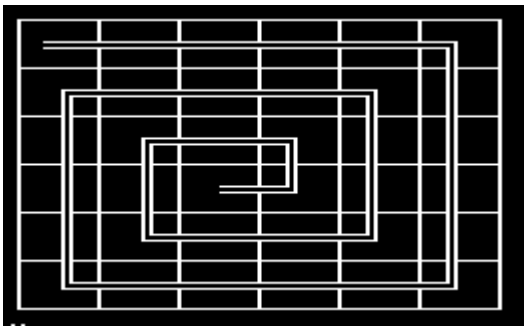

Рис.1. Перемещение по спирали.

4. Содержание отчета по лабораторной работе.

Отчет по лабораторной работе должен содержать :
- формулировку цели и задач;
- описание процесса разработки программ;
- выводы по проделанной реализации.

Литература.

1) Информатика и программирование : шаг за шагом. Язык программирования LISP. // Кафедра Информационных Технологий Курганского Государственного Университета : http://it.kgsu.ru/Lisp/oglav1.html
2) Бородич Ю.С. и др. Паскаль для персональных компьютеров : Справ. пособие. – Мн.: Выш. шк., 1991.
3) Lutz Mueller newLISP™ For BSDs, Linux, Mac OS X, Solaris and Win32. Users Manual and Reference v.9.1 // www.nuevatec.com
4) Tcl / Tk Tool Command Language and Tool Kit - язык программирования и инструментальные средства // http://jarosh.by.ru/tcltkrus/index.htm