Міністерство освіти та науки України

Національний університет «Чернігівська політехніка»

Навчально-науковий інститут електронних та інформаційних технологій

Кафедра інформаційних та комп’ютерних систем

**Модуль «Асемблерна програма для обробки рядків»**

**Програмний проект**

**Листів 10**

Виконав:

студент групи КІ-211ПІБ

Керівник:

професор, д.т.н. Лисенко Д.Е.

2023

Чернігів

**АНОТАЦІЯ**

практичної частини ПІБ на тему: «Асемблерна програма для обробки рядків»

Об'єктом роботи є програма, написана на асемблері, призначена для обробки рядків.

Метою даної роботи є розробка програми, яка виводить рядок на екран у звичайному порядку і записує її в пам'ять у зворотному порядку, а результат виводить на екран.

Розглядається реалізація такої програми, яка може бути використана, де потрібно обробляти рядки.

У ході виконання роботи потрібно:

• описати алгоритм програми та її структуру(псевдокод);

• створити блок-схему програми;

• розробити та реалізувати програму на асемблері для обробки рядків;

• продемонструвати роботу програми.

Виконання роботи проводилось у відповідності з поставленими вимогами.

Результатом роботи є розроблена та протестована програма на асемблері для обробки рядків, що може бути використана, де потрібно обробляти рядки.

**ЗМІСТ**

[1 АНАЛІЗ ОТРИМАНОГО ЗАВДАННЯ 4](#_Toc133432675)

[1.1 Завдання 4](#_Toc133432676)

[1.2 Псевдокод 5](#_Toc133432677)

[1.3 Блок-схема 7](#_Toc133432678)

[2 ВИКОНАННЯ ПРОЕКТУ 8](#_Toc133432679)

[2.1 Код програми 8](#_Toc133432680)

[2.2 Результат виконання програми 11](#_Toc133432681)

[3 ВИСНОВКИ 12](#_Toc133432682)

**1 АНАЛІЗ ОТРИМАНОГО ЗАВДАННЯ**

**1.1 Завдання**

Процедура одержує як параметр адресу sz-рядка, показує його на екрані. Цей же рядок в пам’яти записує в зворотньому порядку и показує його на екрані.

Згідно до завдання треба написати програму, що отримує рядок з нульовим символом в кінці. Програма має інвертувати його й результат має виводитися на екран.

**1.2 Псевдокод**

Псевдокод - це неформальний запис алгоритму, який використовує структуру поширених мов програмування, але нехтує деталями коду, неістотними для розуміння алгоритму (опис типів, виклик підпрограм тощо). Мова програмування доповнюється природною мовою, компактними математичними позначеннями. Псевдокод є зрозумілішим, ніж програми, формою запису алгоритмів.

Псевдокод, що буде наведений нижче стане прообразом програми наведений, відповідно до завдання у лістингу 1.1.

Лістинг 1.1 – Псевдокод

|  |
| --- |
| Declare hInstance, hIcon, tEdit, pbuff, buff as DWORD  entry\_point:  hInstance = GetModuleHandle(0)  hIcon = LoadImage(hInstance, 10, IMAGE\_ICON, 256, 256, LR\_DEFAULTCOLOR)  DialogBoxParam(hInstance, 10, 0, PartsWindow, hIcon)  exit  PartsWindow:  buffer[260] = ""  switch (uMsg)  case WM\_INITDIALOG:  SendMessage(hWin, WM\_SETICON, 1, lParam)  tEdit = GetDlgItem(hWin, 11)  SetFocus(tEdit)  pbuff = buffer  case WM\_COMMAND:  switch (wParam)  case Revers:  GetWindowText(tEdit, pbuff, sizeof buffer)  if (rax == 0)  MessageBox(hWin, "Введіть текст або натисніть Закрити вікно", "Текст не введено", MB\_ICONINFORMATION)  SetFocus(tEdit)  else  rbx = 0  rcx = sizeof buffer  rsi = pbuff  n:  if (byte ptr [rsi] == 0) goto m  rsi = rsi + 1  rbx = rbx + 1  rcx = rcx - 1  goto n  m:  rcx = rbx  rsi = rsi - 1  rdi = buff  loopp:  al = byte ptr [rsi]  byte ptr [rdi] = al  rdi = rdi + 1  rsi = rsi - 1  loop loopp  MessageBox(0, addr buff, "Текст у зворотньому порядку", MB\_ICONINFORMATION)  case Close:  goto exit\_dialog  endswitch  case WM\_CLOSE:  exit\_dialog:  EndDialog(hWin, 0)  endswitch  rax = 0  ret |

**1.3 Блок-схема**

Блок-схема – це представлення алгоритму розв’язання або аналізу задачі за допомогою геометричних елементів (блоків), які позначають операції, потік, дані тощо.

Блок-схема, відповідно до завдання, наведена на рисунку 1.1

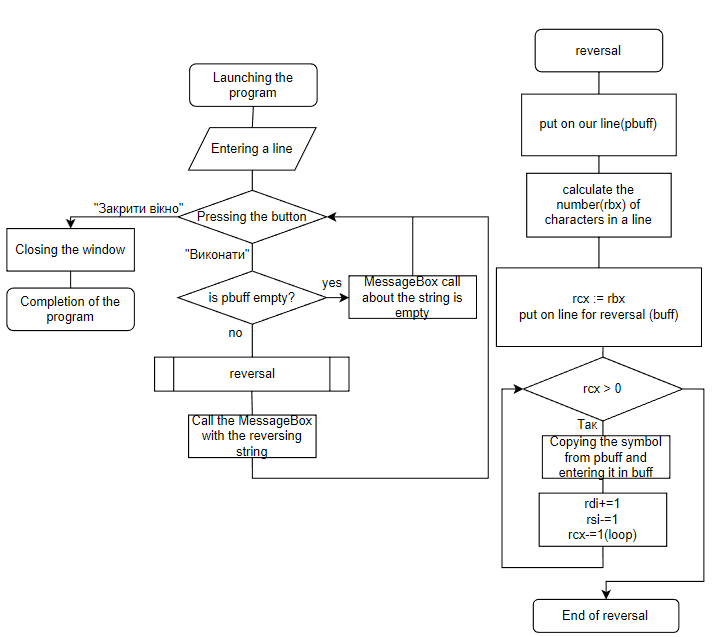


Рисунок 1.1 – Блок-схема програми

**2 ВИКОНАННЯ ПРОЕКТУ**

**2.1 Код програми**

Код програми, відповідно до завдання, наведений в лістингу 2.1

Лістинг 2.1 – Код програми

|  |
| --- |
| include \masm64\include64\masm64rt.inc  .data?  hInstance dq **?**  hIcon dq **?**  tEdit dq **?**  pbuff dq **?**  buff dq **?**    .code  entry\_point proc  **mov** hInstance**,** rvcall**(**GetModuleHandle**,**0**)**  **mov** hIcon**,** rv**(**LoadImage**,**hInstance**,**10**,**IMAGE\_ICON**,**256**,**256**,** LR\_DEFAULTCOLOR**)**  invoke DialogBoxParam**,**hInstance**,**10**,**0**,ADDR** PartsWindow**,**hIcon  .exit  entry\_point endp  PartsWindow proc hWin**:QWORD,**uMsg**:QWORD,**wParam**:QWORD,**lParam**:QWORD**  LOCAL buffer**[**260**]:BYTE**  .switch uMsg  .case WM\_INITDIALOG  invoke SendMessage**,**hWin**,**WM\_SETICON**,**1**,**lParam  **mov** tEdit**,** rvcall**(**GetDlgItem**,**hWin**,**11**)**  invoke SetFocus**,** tEdit  **mov** pbuff**,** ptr**$(**buffer**)**  .case WM\_COMMAND  .switch wParam  .case 12  invoke GetWindowText**,**tEdit**,**pbuff**,**sizeof buffer  .if **rax** **==** 0  rcall MessageBox**,**hWin**,**"Введіть текст або натисніть Закрити вікно"**,**"Текст не введено"**,**MB\_ICONINFORMATION  rcall SetFocus**,**tEdit  .else  **xor** **rbx,rbx**  **mov** **rcx,**sizeof buffer  **mov** **rsi,**pbuff  n**:**  **cmp** **byte** ptr **[rsi],**0  **jz** m  **inc** **rsi**  **inc** **rbx**  **dec** **rcx**  **jnz** n  m**:**  **mov** **rcx,rbx**  **dec** **rsi**  **lea** **rdi,**buff    loopp**:**    **mov** **al,byte** ptr **[rsi]**  **mov** **byte** ptr **[rdi],al**  **inc** **rdi**  **dec** **rsi**  **loop** loopp  invoke MessageBox**,**0**,addr** buff**,**"Текст у зворотньому порядку"**,**MB\_ICONINFORMATION  .endif  .case 13  **jmp** exit\_dialog  .endsw  .case WM\_CLOSE  exit\_dialog**:**  rcall EndDialog**,**hWin**,**0  .endsw  **xor** **rax,** **rax**  **ret**  PartsWindow endp  end |

Rc-файл наведений в лістингу 2.2

Лістинг 2.2 – Rc-файл

|  |
| --- |
| #define IDD\_DLG 10  #define IDC\_Edit 11  #define IDC\_Revers 12  #define IDC\_Close 13  #define IDC\_STC 14  #include "/masm64/include64/resource.h"  10 **ICON** **DISCARDABLE** "icon.ico"  IDD\_DLG **DIALOGEX** 0**,-**12**,**165**,**45  **CAPTION** "Виведення тексту у зворотньому порядку"  **FONT** 10**,**"Segoe UI"**,**400**,**0**,**0  **STYLE** WS\_POPUP**|**WS\_VISIBLE**|**WS\_CAPTION**|**WS\_SYSMENU**|**DS\_CENTER  **BEGIN**  **CONTROL** ""**,**IDC\_Edit**,**"Edit"**,**WS\_CHILDWINDOW**|**WS\_VISIBLE**|**WS\_TABSTOP**,**9**,**12**,**145**,**10**,**WS\_EX\_CLIENTEDGE  **CONTROL** "Виконати"**,**IDC\_Revers**,**"Button"**,**WS\_CHILDWINDOW**|**WS\_VISIBLE**|**WS\_TABSTOP**,**15**,**27**,**50**,**12  **CONTROL** "Закрити вікно"**,**IDC\_Close**,**"Button"**,**WS\_CHILDWINDOW**|**WS\_VISIBLE**|**WS\_TABSTOP**,**99**,**27**,**50**,**12  **CONTROL** "Поле введення"**,**IDC\_STC**,**"Static"**,**WS\_CHILDWINDOW**|**WS\_VISIBLE**,**15**,**3**,**87**,**9  **END** |

**2.2 Результат виконання програми**

На рисунках 2.1 – 2.3 нижче програма запускається декілька разів:

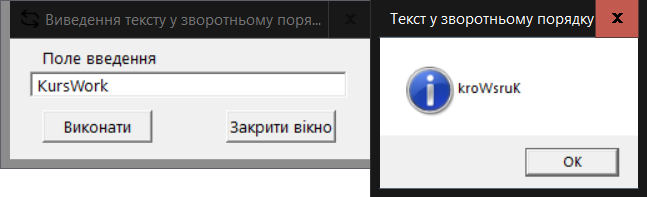


Рисунок 2.1 – Виконання програми анлійськими літерами

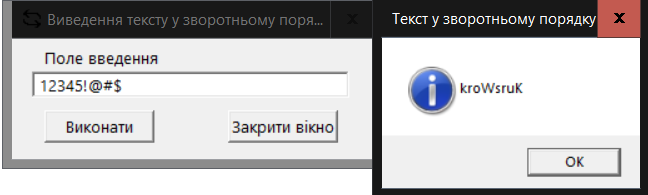


Рисунок 2.2 – Виконання програми числами та знаками

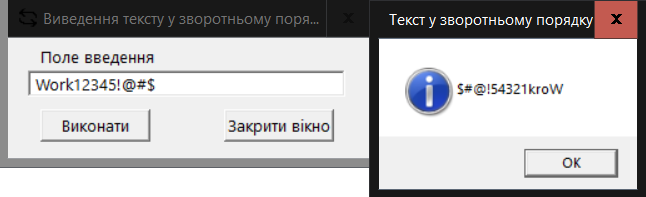


Рисунок 2.3 – Виконання програми з літерами, числами та знаками

**3 ВИСНОВКИ**

В ході виконання курсової роботи була досягнута основна її мета – розробити програму, написану на асемблері, яка друкує рядок на екран у звичайному порядку, записує його в пам’ять у зворотному порядку та відображає результат на екрані.

Згідно з поставленою задачею було описано алгоритм програми та її структуру, створено блок-схему для більшого розуміння та простішого виконання завдання.

Програма має діалогове вікно з текстовим полем і двома кнопками. Можна ввести текст у текстове поле, натиснути кнопку «Виконати», щоб змінити текст у зворотному напрямку та відобразити його у вікні, або натиснути кнопку «Закрити вікно», щоб закрити вікно.

Правильність роботи здійснювалася шляхом запуску програми і кожного разу введення різних символів.

Отже, можна зробити висновок, що мета курсової роботи була досягнута успішно. Було розроблено програму на асемблері, яка здійснює обробку рядків, та проведено тестування її роботи.