МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

«ХАРКІВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра «Програмна інженерія та інформаційні технології управління»

Звіт з лабораторної роботи №7

З предмету «Операційні системи»

Виконав

Студент групи КН-36а

Рубан Ю.Д.

Перевірив:

Кондратов О.М.

Харків 2017

ВВЕДЕНИЕ В МЕЖПРОЦЕССНОЕ ВЗАИМОДЕЙСТВИЕ. СОКЕТЫ.

**Последовательность выполнения работы**

1. Ознакомиться теоретическим материалом.

2. Получить у преподавателя индивидуальный вариант задания, который должен предусматривать разработку двух программ (сервер и клиент) и взаимодействие между ними посредством сокетов.

3. Разработать и отладить сервер и клиент в соответствии с полученным заданием.

4. Написать отчет и представить его к защите вместе с исполняемыми модулями программ и их исходными текстами.

**Задание на лабораторную работу:**

Разработать две программы – сервер и клиент. Клиент отсылает серверу через датаграммный сокет две строки, введенные пользователем. Сервер принимает из именованного конвейера две строки, осуществляет поиск вхождения второй строки в первую любым известным методом, кроме прямого (алгоритм Кнута-Мориса-Пратта, алгоритм Боуэра-Мура), и выводит на экран значение индекса элемента первой строки, с которого началось совпадение, или -1 в противном случае.

**Код программы:**

Код сервера:

#include<iostream>

#include<string>

#include<vector>

#include<string.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<unistd.h>

using namespace std;

int max(const int&a, const int&b);

int\*\* getRes(int \*\*res, string x, string y);

string getSubSequence(int \*\*res, string x, string y);

void getArrows(string x, string y);

string\* split(string str);

int main()

{

unlink("sock.soc");

int soc = socket(AF\_UNIX,SOCK\_DGRAM,0);

struct sockaddr srvr\_name,rcvr\_name;

socklen\_t namelen;

srvr\_name.sa\_family = AF\_UNIX;

strcpy(srvr\_name.sa\_data,"sock.soc ");

bind(soc,&srvr\_name,strlen(srvr\_name.sa\_data));

string X="";

string Y="";

char\* buffer;

buffer = new char[1024];

int bytes = recvfrom(soc,buffer,1024,0,NULL,NULL);

cout<<buffer<<endl;

string buff = buffer;

string\* xy = split(buff);

X = xy[0];

Y = xy[1];

close(soc);

int n = X.size();

int m = Y.size();

int \*\*res = new int\*[n + 1];

for (int i = 0; i<n + 1; i++)

{

res[i] = new int[m + 1];

}

for (int i = 0; i < n + 1; i++)

{

for (int j = 0; j < m + 1; j++)

{

res[i][j] = 0;

}

}

cout << "Dynamic table :" << endl;

res = getRes(res, Y, X);

string sub = getSubSequence(res, Y, X);

cout << "Arrows: " << endl;

getArrows(Y, X);

cout << "Sub sequence" << endl;

for (int i = 0; i < sub.size(); i++)

{

cout << sub[i] << " ";

}

cout << endl;

return 0;

}

int max(const int&a, const int&b) { return a > b ? a : b; }

int\*\* getRes(int \*\*res, string x, string y)

{

int n = x.size();

int m = y.size();

cout << " ";

for (int i = 0; i<y.size(); i++)

{

cout << " " <<y[i];

}

cout << endl;

for (int i = 1; i <=n; i++)

{

cout << x[i-1] << " ";

for (int j = 1; j <=m; j++)

{

if (x[i-1] == '\0' || y[j-1] == '\0') { res[i][j] = 0; }

else if (x[i-1] == y[j-1]) { res[i][j] = res[i - 1][j - 1] + 1; }

else

{

res[i][j] = max(res[i - 1][j], res[i][j - 1]);

}

cout << res[i][j] << " ";

}

cout << endl;

}

return res;

}

string\* split(string str)

{

vector<string>res;

char \* pch;

char\*temp = new char[str.size()-1];

strcpy(temp,str.c\_str());

pch = strtok (temp," ,.-\n");

while (pch != NULL)

{

res.push\_back(pch);

pch = strtok (NULL, " ,.-");

}

string\* r = new string[res.size()];

for(int i =0;i<res.size();i++)

{

r[i]=res[i];

}

return r;

}

string getSubSequence(int \*\*res, string x,string y)

{

int n = x.size();

int m = y.size();

int size = res[m][n];

int i = x.size()-1;

int j = y.size()-1;

string sub;

int ik, jk;

ik = i + 1;

jk = j + 1;

while (i >= 0 && j >= 0)

{

if (x[i] == y[j])

{

sub += x[i];

i--; j--;

ik--;

jk--;

}

else if (res[ik - 1][jk] >= res[ik][jk - 1]) { i--; ik--; }

else { j--; jk--; }

}

string result;

for (int i = sub.size() - 1; i >= 0; i--)

{

result += sub[i];

}

return result;

}

void getArrows(string x, string y)

{

int n = x.size();

int m = y.size();

int\*\*res = new int\*[n + 1];

for (int i = 0; i < n + 1; i++)

{

res[i] = new int[m + 1];

}

for (int i = 0; i < n + 1; i++)

{

for (int j = 0; j < m + 1; j++)

{

res[i][j] = 0;

}

}

vector<string>arrows(n+1);

for (int i = 0; i < n+1; i++)

{

arrows[i].resize(m+1);

}

int ik=0, jk=0;

cout << " ";

for (int i = 0; i<y.size(); i++)

{

cout << " " << y[i];

}

cout << endl;

for (int i = 1; i <= n; i++)

{

cout << x[i - 1] << " ";

for (int j = 1; j <= m; j++)

{

if (x[i - 1] == '\0' || y[j - 1] == '\0') { res[i][j] = 0; }

else if (x[i - 1] == y[j - 1]) { res[i][j] = res[i - 1][j - 1] + 1; arrows[ik][jk] += 'd';}

else if(res[i - 1][j]>=res[i][j - 1])

{

res[i][j] = res[i - 1][j];

arrows[ik][jk] += 'u';

}

else

{

res[i][j] = res[i ][j- 1];

arrows[ik][jk] += 'l';

}

cout << arrows[ik][jk] << " ";

jk++;

}

ik++;

jk = 0;

cout << endl;

}

}

Код клиента:

#include<stdio.h>

#include<cstdio>

#include<iostream>

#include<string>

#include<string.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/socket.h>

using namespace std;

int main()

{

int soc = socket(AF\_UNIX,SOCK\_DGRAM,0);

struct sockaddr srvr\_name;

srvr\_name.sa\_family = AF\_UNIX;

strcpy(srvr\_name.sa\_data,"sock.soc");

printf("enter 2 strings\n");

string x,y;

cin>>x;

x+="\n";

cin>>y;

x+=y;

char\*buffer = new char[100];

strcpy(buffer,x.c\_str());

sendto(soc,buffer,1024,0,&srvr\_name,strlen(srvr\_name.sa\_data)+sizeof(srvr\_name.sa\_family));

close(soc);

return 0;

}

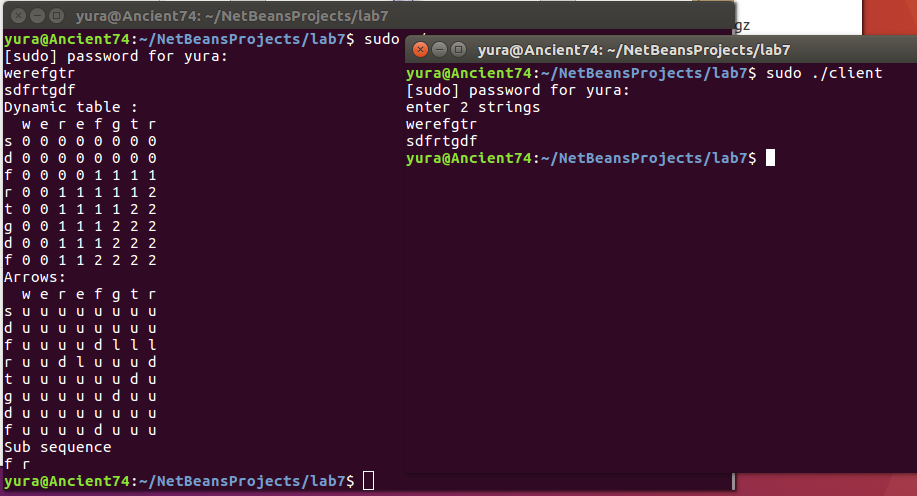


Рисунок 1.1 Результаты выполнения программы

**Выводы:** В данной лабораторной работе был изучен теоретический материал по межпроцессорному программированию и сокетам.