LINUX实验报告

硬件一班

翟博

1. 实验目的

掌握Linux一般命令格式

掌握有关文件和目录操作的常用命令

掌握有关进程操作的常用命令

1. 实验内容

在用户主目录下对文件进行操作

对目录进行管理

1. 主要实验步骤

#include<stdio.h>

#include<stdlib.h>

#include<malloc.h>

#include<string.h>

#include<Windows.h>

#define STACK\_SIZE 6

void Empty(struct Stack \*temp);/\*创建一个空栈的辅助函数\*/

struct Stack \*CreateStack(void);/\*创建一个空栈\*/

int IsEmpty(struct Stack \*S);/\*测试栈是否为空\*/

void Push( struct Stack\*sp); /\*进栈例程\*/

int Top(struct Stack\*temp); /\*返回栈顶元素\*/

void Pop(struct Stack\*temp); /\*从栈弹出元素\*/

void Rebuild(struct Stack \*p); /\*释放栈空间\*/

void Print(); /\*菜单打印\*/

void Exit(); /\*退出\*/

struct Stack { /\*栈的声明\*/

int Element;

struct Stack \*next;

};

struct Stack \*stack = NULL;

int main(void) {

char ch = '\0';

while (1) {

system("cls");

fflush(stdin);

Print();

scanf(" %c",&ch);

switch (ch) {

case'a':

stack=CreateStack();

break;

case'b':

Push(stack);

break;

case'c':

Top(stack);

break;

case'd':

Pop(stack);

break;

case'e':

Rebuild(stack);

break;

case'f':

Exit();

break;

default:

printf("无此选择项!\n");

system("pause");

break;

}

}

return 0;

}

void Rebuild(struct Stack \*p) {

if (p==NULL) {

printf("Must use CreateStack first!");

return;

}

struct Stack \*temp = NULL;

struct Stack \*tt = NULL;

temp = p;

p->next = NULL;

while (temp != NULL) {

tt = temp->next;

free(temp);

temp = tt;

}

puts("Rebuild!\n");

system("pause");

}

void Exit() { /\*这个是用来退出\*/

int i = 0;

printf("退出中");

for (i = 4; i > 0; --i) {

Sleep(200);

printf(".");

}

exit(0);

}

void Print()

{

printf("-----主菜单功能如下:\n");

printf("-----a.创建一个栈\n");

printf("-----b.Push进栈\n");

printf("-----c.Top返回栈顶元素\n");

printf("-----d.Pop出栈\n");

printf("-----e.释放栈空间\n");

printf("-----f.退出\n");

}

int IsEmpty(struct Stack \*S) {

return S->next == NULL?1:0;

}

struct Stack \*CreateStack(void){

struct Stack \*temp=NULL;

struct Stack \*current = NULL;

struct Stack \*last = NULL;

int size = STACK\_SIZE;

int a = 0;

temp = (struct Stack\*)malloc(sizeof(struct Stack));

while (size-- > 0)

{

current = (struct Stack\*)malloc(sizeof(struct Stack));

if (temp == NULL)

temp = current;

if (last != NULL)

last->next = current;

current->next = NULL;

last = current;

}

puts("Created!");

return (temp);

}

void Empty(struct Stack \*temp) {

if (temp == NULL) {

printf("Must use CreateStack first!");

return;

}

else

temp->next = NULL;

}

void Push( struct Stack\*sp)

{

int x=0;

struct Stack \*temp = NULL;

temp = (struct Stack\*)malloc(sizeof(struct Stack));

if (temp == NULL) {

printf( "out of space!");

}

else {

printf("Please enter a Element x!\n");

scanf("%d",&x);

temp->Element = x;

temp->next = sp->next;

sp->next = temp;

}

}

int Top(struct Stack\*temp) {

if (!IsEmpty(temp))

return temp->next->Element;

printf( "Empty stack");

return 0;

}

void Pop(struct Stack\*temp) {

struct Stack \*first = NULL;

if(IsEmpty(temp))

printf("Empty stack");

else

{

first = temp->next;

temp->next = temp->next->next;

free(first);

}

}

1. 实验总结

通过这次实验，掌握了linux一般命令格式，希望通过学习能更加了解这门学科。