ARYAMAN MISHRA

19BCE1027

Dr.B.Saleena

1)

#include <stdio.h>

#include <string.h>

#include <ctype.h>

#include <stdlib.h>

struct Stack

{

int top;

unsigned capacity;

int\* array;

};

struct Stack\* createStack( unsigned capacity )

{

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

if (!stack) return NULL;

stack->top = -1;

stack->capacity = capacity;

stack->array = (int\*) malloc(stack->capacity \* sizeof(int));

if (!stack->array) return NULL;

return stack;

}

int isEmpty(struct Stack\* stack)

{

return stack->top == -1 ;

}

char peek(struct Stack\* stack)

{

return stack->array[stack->top];

}

char pop(struct Stack\* stack)

{

if (!isEmpty(stack))

return stack->array[stack->top--] ;

return '$';

}

void push(struct Stack\* stack, char op)

{

stack->array[++stack->top] = op;

}

int evaluatePostfix(char\* exp)

{

struct Stack\* stack = createStack(strlen(exp));

int i;

if (!stack) return -1;

for (i = 0; exp[i]; ++i)

{

if (isdigit(exp[i]))

push(stack, exp[i] - '0');

else

{

int val1 = pop(stack);

int val2 = pop(stack);

switch (exp[i])

{

case '+': push(stack, val2 + val1); break;

case '-': push(stack, val2 - val1); break;

case '\*': push(stack, val2 \* val1); break;

case '/': push(stack, val2/val1); break;

}

}

}

return pop(stack);

}

int main()

{

char a,b,c,d;

printf("Enter values for postfix expression A B C + \* D\* \n");

scanf(" %c",&a);

scanf(" %c",&b);

scanf(" %c",&c);

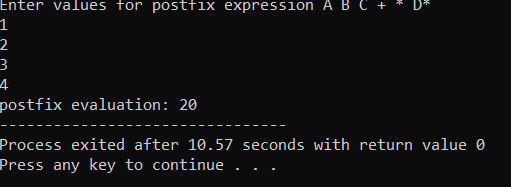
scanf(" %c",&d);

char exp[]={a,b,c,'+','\*',d,'\*'};

printf ("postfix evaluation: %d", evaluatePostfix(exp));

return 0;

}



2) #include <stdio.h>

#include<stdlib.h>

#define MAX 10

char queue\_array[MAX];

int rear = - 1;

int front = - 1;

void insert()

{

char program;

if(rear == MAX - 1)

printf("Queue Overflow \n");

else

{

if(front== - 1)

front = 0;

printf("Inset the program in queue : ");

scanf(" %c", &program);

rear = rear + 1;

queue\_array[rear] = program;

}

}

void dequeue()

{

if(front == - 1 || front > rear)

{

printf("Queue Underflow \n");

return;

}

else

{

printf("Element deleted from queue is : %c \n", queue\_array[front]);

front = front + 1;

}

}

void display()

{

int i;

if(front == - 1)

printf("Queue is empty \n");

else

{

printf("Queue is : \n");

for(i = front; i <= rear; i++)

printf("%c ", queue\_array[i]);

printf("\n");

}

}

int main()

{

int count=0,choice;

while (count<=60)

{

printf("1.Insert program to printer\n");

printf("2.Print Program from Printer\n");

printf("3.Display waiting programs to printer\n");

printf("4.Quit \n");

printf("Enter your choice : ");

scanf("%d", &choice);

switch(choice)

{

case 1:

insert();

count++;

break;

case 2:

dequeue();

break;

case 3:

display();

break;

case 4:

exit(1);

default:

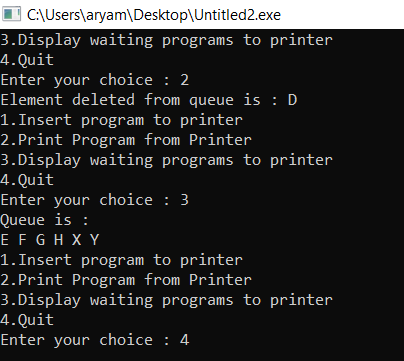
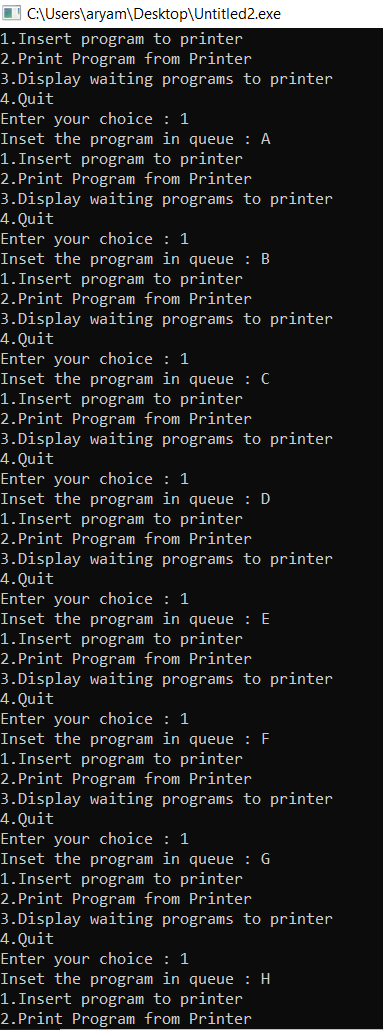
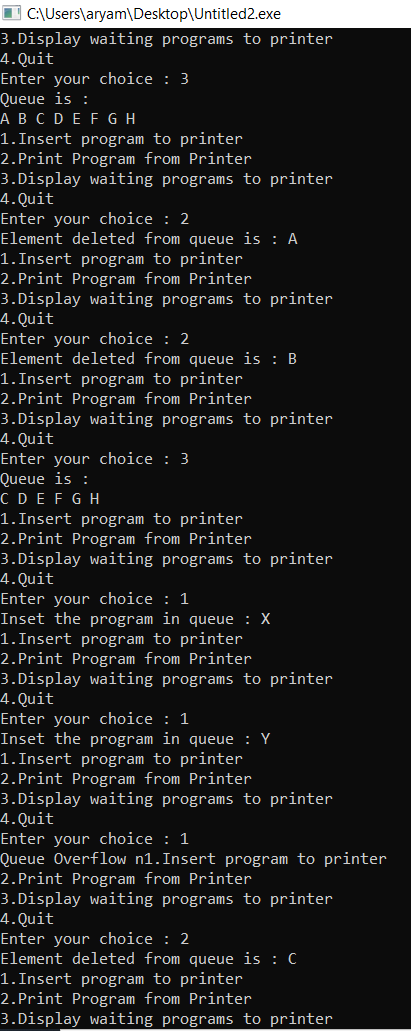
printf("Wrong choice n");

return 0;

}

}

}



3)

#include <stdio.h>

#include<stdlib.h>

#define MAX 1000

void business\_insert();

void normal\_insert();

void normal\_dequeue();

void business\_dequeue();

void display();

char normal\_queue\_array[MAX];

char business\_queue\_array[MAX];

int brear = - 1;

int bfront = - 1;

int nrear = - 1;

int nfront = - 1;

int n\_count=0;

int b\_count=0;

void business\_insert()

{

char item;

if(brear == MAX - 1)

printf("Queue Overflow n");

else

{

if(bfront== - 1)

bfront = 0;

printf("Inset the element in queue : ");

scanf(" %c", &item);

brear = brear + 1;

business\_queue\_array[brear] = item;

b\_count++;

}

}

void normal\_insert()

{

char item;

if(nrear == MAX - 1)

printf("Queue Overflow n");

else

{

if(nfront== - 1)

nfront = 0;

printf("Inset the element in queue : ");

scanf(" %c", &item);

nrear = nrear + 1;

normal\_queue\_array[nrear] = item;

n\_count++;

}

}

void normal\_dequeue()

{

if(nfront == - 1 || nfront > nrear)

{

printf("Queue Underflow n");

return;

}

else

{

printf("Element deleted from queue is : %c\n", normal\_queue\_array[nfront]);

nfront = nfront + 1;

}

}

void business\_dequeue()

{

if(bfront == - 1 || bfront > brear)

{

printf("Queue Underflow n");

return;

}

else

{

printf("Element deleted from queue is : %c\n", business\_queue\_array[bfront]);

bfront = bfront + 1;

}

}

void ndisplay()

{

int i;

if(nfront == - 1)

printf(" Normal Queue is empty \n");

else

{

printf("Normal Queue is : \n");

for(i = nfront; i <= nrear; i++)

printf(" %c", normal\_queue\_array[i]);

printf("\n");

}

}

void bdisplay()

{

int i;

if(bfront == - 1)

printf("Business Queue is empty \n");

else

{

printf("Business Queue is : \n");

for(i = bfront; i <= brear; i++)

printf(" %c", business\_queue\_array[i]);

printf("\n");

}

}

int main()

{

int choice;

printf("\*\*\*\*\*\*\*\*\*Welcome to EA Airlines\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\*\*\*\*\*\*\*\*\*\*\*It's in the fLight\*\*\*\*\*\*\*\*\*\*\*\n");

while (1)

{

printf("1.Enter people in Business Class\n");

printf("2.Enter people in Normal Class\n");

printf("3.Quit \n");

printf("Enter your choice : ");

scanf("%d", &choice);

if(choice==3)

break;

switch(choice)

{

case 1:

business\_insert();

break;

case 2:

normal\_insert();

break;

case 3:

break;

default:

printf("Wrong choice \n");

}

}

if(b\_count==n\_count)

{

while(1)

{

printf("1.Allow passengers to go through from queue \n");

printf("2.Display standing passengers in queue \n");

printf("3.Quit \n");

printf("Enter your choice: \n");

scanf("%d",&choice);

if(choice==3)

break;

switch(choice){

case 1:normal\_dequeue();

business\_dequeue();

break;

case 2:bdisplay();

ndisplay();

break;

case 3:

exit(1);

default:printf("Wrong Choice");

}

}

}

if(n\_count<(b\_count/2))

{

while(1)

{

printf("1.Allow passengers to go through from queue \n");

printf("2.Display standing passengers in queue \n");

printf("3.Quit \n");

printf("Enter your choice: \n");

scanf("%d",&choice);

if(choice==3)

break;

switch(choice){

case 1:business\_dequeue();

business\_dequeue();

normal\_dequeue();

break;

case 2:bdisplay();

ndisplay();

break;

case 3:exit(1);

default:printf("Wrong Choice");

}

}

}

}

