

Introducción a la Programación I – Práctico 2 (Filas)

```
Program e1p2;  
{paso los elementos de la fila origen, a la pila destino}  
uses estructu;  
var origen:fila;  
    destino:pila;  
begin  
    readfila(origen);  
    inicpila(destino, '');  
    while not filavacia (origen) do  
        apilar(destino, extraer(origen));  
    writepila(destino);  
end.  
  
/*****/  
  
Program e3p2;  
{invierte el orden de la fila dada}  
uses estructu;  
var dada:fila;  
    aux:pila;  
begin  
    readfila(dada);  
    inicpila(aux, '');  
    while not filavacia (dada) do  
        apilar(aux, extraer(dada));  
    while not pilavacia (aux) do  
        agregar(dada, desapilar(aux));  
    writefila(dada);  
end.  
  
/*****/  
  
Program e4p2;  
{pasa el primer elemento de la fila datos, a su ultima posicion}  
uses estructu;  
var datos:fila;  
begin  
    readfila(datos);  
    if not filavacia (datos) then  
        agregar(datos, extraer(datos));  
    writefila(datos);  
end.  
  
/*****/
```

```
Program e5p2;
{pasa el ultimo elemento de la fila datos, a su primer posicion}
uses estructu;
var datos:fila;
    aux1,aux2:pila;
begin
    readfila(datos);
    inicpila(aux1, '');
    inicpila(aux2, '');
    while not filavacia (datos) do
        apilar(aux1, extraer(datos));
    if not pilavacia (aux1) then
        agregar(datos, desapilar(aux1));
    while not pilavacia (aux1) do
        apilar(aux2, desapilar(aux1));
    while not pilavacia (aux2) do
        agregar(datos, desapilar(aux2));
    writefila(datos);
end.
/*****/

Program e2p6;
{imprime por pantalla el el resultado de un numero elevado a otro}

function Multiplica(primer,ultimo:integer):integer;
{obtiene el valor de la multiplicacion de 2 numeros usando sumas}
var auxiliar:integer;
begin
    auxiliar:=0;
    while (primer > 0) do
        begin
            auxiliar:=auxiliar + ultimo;
            primer:=primer-1;
        end;
    Multiplica:=auxiliar;
end;

var a,b,resultado:integer;
begin
    write('ingrese un numero ');
    read(a);
    write('ingrese otro numero este, sera el exponente del anterior ');
    read(b);
    resultado:=a;
    while (b>1) do
        begin
            resultado:=multiplica(a,resultado);
            b:=b-1;
        end;
    writeln('resultado = ',(resultado));
end.
```

```
Program e7p2;  
{imprime la pila1 o la pila2, segun un valor ingresado por el usuario}  
uses estructu;  
var pila1,pila2,valor:pila;  
begin  
  inicpila(pila1,'1 2 3 4 5 6');  
  inicpila(pila2,'1 1 1 4 5 6 7 8 4 5 4');  
  readpila(valor);  
  if (tope(valor))= 1 then  
    writepila(pila1);  
  if (tope(valor))= 2 then  
    writepila(pila2);  
end.  
  
/*****/
```

```
Program e8p2;  
{pone los elementos de la pila origen, que son iguales al tope de la  
pila modelo, en su ultima posicion}  
uses estructu;  
var modelo,origen,aux1,aux2:pila;  
begin  
  inicpila(origen,'1 2 3 4 5 6 7 8 9 2 3 2 4 5 23 5');  
  readpila(modelo);  
  inicpila(aux1,'');  
  inicpila(aux2,'');  
  while not pilavacia (origen) and not pilavacia (modelo) do  
    if (tope (modelo)) = (tope (origen)) then  
      apilar(aux1, desapilar(origen))  
    else  
      apilar(aux2, desapilar(origen));  
  while not pilavacia (aux2) do  
    apilar(origen, desapilar(aux2));  
  while not pilavacia (aux1) do  
    apilar(origen, desapilar(aux1));  
  writepila(origen);  
end.  
  
/*****/
```

```
Program e9p2;  
{ubica el tope de la pila dada, debajo del primer elemento de valor 12}  
uses estructu;  
var dada,aux1,aux2:pila;  
begin  
  readpila(dada);  
  inicpila(aux1, '');  
  inicpila(aux2, '');  
  if not pilavacia (dada) and  
    (tope(dada) <> 12) then  
    apilar(aux1, desapilar(dada));  
  while not pilavacia (dada) and  
    (tope(dada) <> 12) do  
    apilar(aux2, desapilar(dada));  
  if not pilavacia (dada) and  
    (tope(dada) = 12) then  
    apilar(aux2, desapilar(dada));  
  if not pilavacia (aux1) then  
    apilar(dada, desapilar(aux1));  
  while not pilavacia (aux2) do  
    apilar(dada, desapilar(aux2));  
  writepila(dada);  
end.
```

```
/*****/
```

```
Program e10p2;  
{Junta en una pila otras 2 pilas, de modo que la que tiene menos  
elementos quede debajo}  
uses estructu;  
var origen1,origen2,destino,aux1,aux2:pila;  
begin  
  readpila(origen1);  
  readpila(origen2);  
  inicpila(destino, '');  
  inicpila(aux1, '');  
  inicpila(aux2, '');  
  while (not pilavacia(origen1)) and (not pilavacia(origen2)) do  
    begin  
      apilar(aux1, desapilar(origen1));  
      apilar(aux2, desapilar(origen2));  
    end;  
  if pilavacia(origen1) and pilavacia(origen2) then  
    begin  
      while (not pilavacia(aux1)) and (not pilavacia(aux2)) do  
        begin  
          apilar(origen1, desapilar(aux1));  
          apilar(origen2, desapilar(aux2));  
        end;  
      while (not pilavacia(origen1)) do
```

```
    apilar(destino, desapilar(origen1));
    while (not pilavacia(origen2)) do
        apilar(destino, desapilar(origen2));
    end;
if (pilavacia(origen1)) and (not pilavacia(origen2)) then
    begin
        while (not pilavacia(aux1)) and (not pilavacia(aux2)) do
            begin
                apilar(origen1, desapilar(aux1));
                apilar(origen2, desapilar(aux2));
            end;
        while (not pilavacia(origen1)) do
            apilar(destino, desapilar(origen1));
        while (not pilavacia(origen2)) do
            apilar(destino, desapilar(origen2));
        end;
    if (not pilavacia(origen1)) and pilavacia(origen2) then
        begin
            while (not pilavacia(aux1)) and (not pilavacia(aux2)) do
                begin
                    apilar(origen1, desapilar(aux1));
                    apilar(origen2, desapilar(aux2));
                end;
            while (not pilavacia(origen2)) do
                apilar(destino, desapilar(origen2));
            while (not pilavacia(origen1)) do
                apilar(destino, desapilar(origen1));
            end;
        writepila(destino);
    end.

/*****/
```

```
program e11p2;
{ordenamos la pila copia, para que quede en el mismo orden que la pila
original}
uses estructu;
var original, copia, aux1, aux2, aux:pila;
begin
  readpila(original);
  write('ingrese los mismos datos que antes, pero en distinto orden');
  readpila(copia);
  inicpila(aux, '');
  inicpila(aux1, '');
  inicpila(aux2, '');
  while not pilavacia(original) do
    begin
      while not pilavacia(copia) do
        if (tope(original) = (tope(copia)) then
          apilar(aux1, desapilar(copia))
        else
          apilar(aux2, desapilar(copia));
        while not pilavacia(aux2) do
          apilar(copia, desapilar(aux2));
          apilar(aux, desapilar(original));
        end;
      while not pilavacia(aux) do
        apilar(original, desapilar(aux));
      while not pilavacia (aux1) do
        apilar(copia, desapilar(aux1));
      writepila(original);
      writepila(copia);
    end.

  /*****/
```

```
program e12p2;  
uses estructu;  
var parte, grande, aux, aux1, aux2:pila;  
begin  
  readpila(parte);  
  readpila(grande);  
  inicpila(aux, '');  
  inicpila(aux1, '');  
  inicpila(aux2, '');  
  while not pilavacia(grande) and (not pilavacia(parte)) do  
    begin  
      if tope(grande)=(tope(parte)) then  
        begin  
          apilar(aux, desapilar(grande));  
          apilar(aux1, desapilar(parte));  
        end  
      else  
        begin  
          apilar(aux, desapilar(grande));  
          while not pilavacia(aux1) do  
            apilar(parte, desapilar(aux1));  
          end;  
        end;  
      writepila(parte);  
    end.  
  
  /*****/
```

```
program e13p2;
{inserte un elemento en la fila dada y esta continua ordenada}
uses estructu;
var elemento:pila;
    dada, auxdada, aux:fila;
begin
    readfila(dada);
    readpila(elemento);
    inicfila(aux, '');
    inicfila(auxdada, '');
    if not filavacia(dada) then
        agregar(auxdada, extraer(dada));
    if not filavacia(auxdada) and ((primero(dada)) < (primero(auxdada))) then
        begin
            while not filavacia(dada) do
                agregar(auxdada, extraer(dada));
            while not filavacia(auxdada) do
                agregar(dada, extraer(auxdada));
            while not filavacia(dada) and (not
pilavacia(elemento)) and (primero(dada) > (tope(elemento))) do
                agregar(aux, extraer(dada));
            if not pilavacia(elemento) then
                agregar(aux, desapilar(elemento));
            while not filavacia(dada) do
                agregar(aux, extraer(dada));
            while not filavacia(aux) do
                agregar(dada, extraer(aux));
        end;
    if not filavacia(auxdada) and ((primero(dada)) > (primero(auxdada))) then
        begin
            while not filavacia(dada) do
                agregar(auxdada, extraer(dada));
            while not filavacia(auxdada) do
                agregar(dada, extraer(auxdada));
            while not filavacia(dada) and (not
pilavacia(elemento)) and (primero(dada) < (tope(elemento))) do
                agregar(aux, extraer(dada));
            if not pilavacia(elemento) then
                agregar(aux, desapilar(elemento));
            while not filavacia(dada) do
                agregar(aux, extraer(dada));
            while not filavacia(aux) do
                agregar(dada, extraer(aux));
        end;
    if filavacia(dada) and (not pilavacia(elemento)) then
        agregar(dada, desapilar(elemento));
    writefila(dada);
end.
```