

## Introducción a la Programación I – Práctico 4 (Procedimientos)

```
Program E1P4A;
{Pasa los elementos de la pila origen a la pila destino, pero dejandolos en el mismo orden}
uses estructu;
var origen, destino, aux:pila;

procedure Pasapila (var primera:pila;var ultima:pila);
{Pasa todos los elementos de primera a ultima}
begin
  while not pilavacia (primera) do
    apilar (ultima, desapilar (primera));
end;

begin
  readpila (origen);
  inicpila (destino, '');
  inicpila (aux, '');
  Pasapila (origen, aux);
  Pasapila (aux, destino);
  writepila (destino);
end.

/*****/

Program E1P4B;
{Invierte el orden de una pila dada}
uses estructu;
var dada, aux1, aux2:pila;

procedure Pasapila (var primera:pila;var ultima:pila);
{Pasa todos los elementos de primera a ultima}
begin
  while not pilavacia (primera) do
    apilar (ultima, desapilar (primera));
end;

begin
  readpila (dada);
  inicpila (aux1, '');
  inicpila (aux2, '');
  Pasapila (dada, aux1);
  Pasapila (aux1, aux2);
  Pasapila (aux2, dada);
  writepila (dada);
end.

/*****/
```

```
Program E2P4A;
{Elimina de la pila dada todos los elementos iguales al tope de la pila
modelo}
uses estructu;
var modelo,dada:pila;

procedure Eliminarrepetidos (var primera:pila;var ultima:pila);
{elimina de la pila primera todos los elementos iguales al tope de la
pila ultima}
var aux,descarte:pila;
begin
  inicpila(aux,'');
  inicpila(descarte,'');
  while not pilavacia(primer) do
    if (tope(primer)=tope(ultima)) then
      apilar(descarte, desapilar(primer))
    else
      apilar(aux, desapilar(primer));
  while not pilavacia(aux) do
    apilar(primer, desapilar(aux));
end;

begin
  readpila(dada);
  inicpila(modelo,'1 2 3 5');
  Eliminarrepetidos(dada,modelo);
  writepila(dada);
end.

/*****/
```

```
Program E2P4B;
{Elimina de la pila dada todos los elementos existentes la pila modelo}
uses estructu;
var modelo,dada,auxiliar:pila;

procedure Eliminarrepetidos (var primera:pila;var ultima:pila);
{elimina de la pila primera todos los elementos iguales al tope de la
pila ultima}
var aux,descarte:pila;
begin
  inicpila(aux, '');
  inicpila(descarte, '');
  while not pilavacia (primera) do
    if (tope (primera)=tope (ultima)) then
      apilar (descarte, desapilar (primera))
    else
      apilar (aux, desapilar (primera));
    while not pilavacia (aux) do
      apilar (primera, desapilar (aux));
  end;

begin
  readpila (dada);
  readpila (modelo);
  inicpila (auxiliar, '');
  while not pilavacia (modelo) do
    begin
      Eliminarrepetidos (dada,modelo);
      apilar (auxiliar, desapilar (modelo));
    end;
    while not pilavacia (auxiliar) do
      apilar (modelo, desapilar (auxiliar));
  writepila (dada);
  writepila (modelo);
end.

/*****/
```

```
Program E2P4C;
{Elimina los elementos repetidos de una pila dada}
uses estructu;
var dada,auxiliar:pila;

procedure Eliminarrepetidos (var primera:pila;var ultima:pila);
{elimina de la pila primera todos los elementos iguales al tope de la
pila ultima}
var aux,descarte:pila;
begin
  inicpila(aux, '');
  inicpila(descarte, '');
  while not pilavacia (primera) do
    if (tope (primera)=tope (ultima)) then
      apilar (descarte, desapilar (primera))
    else
      apilar (aux, desapilar (primera));
    while not pilavacia (aux) do
      apilar (primera, desapilar (aux));
  end;

begin
  readpila (dada);
  inicpila (auxiliar, '');
  while not pilavacia (dada) do
    begin
      apilar (auxiliar, desapilar (dada));
      Eliminarrepetidos (dada, auxiliar);
    end;
    while not pilavacia (auxiliar) do
      apilar (dada, desapilar (auxiliar));
  writepila (dada);
end.

/*****/
```

```
Program E3P4;
{Ordena la pila copia, para que quede igual que original}
uses estructu;
var original, copia, auxcopia, auxoriginal:pila;

procedure Igualatopes (var primera:pila;ultima:pila);
{deja en el tope de inicial el mismo elemento que esta en el tope de
final}
var aux1,aux2:pila;
begin
  inicpila(aux1, '');
  inicpila(aux2, '');
  while not pilavacia (primera) do
    if (tope (primera)=tope (ultima)) then
      apilar(aux1, desapilar(primera))
    else
      apilar(aux2, desapilar(primera));
  while not pilavacia (aux2) do
    apilar(primera, desapilar(aux2));
  while not pilavacia (aux1) do
    apilar(primera, desapilar(aux1));
end;

begin
  readpila(original);
  readpila(copia);
  inicpila(auxoriginal, '');
  inicpila(auxcopia, '');
  while (not pilavacia (original) and not pilavacia (copia)) do
    begin
      Igualatopes (copia,original);
      apilar(auxcopia, desapilar(copia));
      apilar(auxoriginal, desapilar(original));
    end;
  while not pilavacia (auxcopia) do
    apilar(copia, desapilar(auxcopia));
  while not pilavacia (auxoriginal) do
    apilar(original, desapilar(auxoriginal));
  writepila(copia);
  writepila(original);
end.

/*****/
```

```
Program E4P4;
{Ordena la fila dada de forma descendente}
uses estructu;
var dada,mayor:fila;

procedure Sacamayor(var inicial,ultimo:fila);
{separa el mayor elemento de la fila inicial}
var aux1,aux2:pila;
begin
  inicpila(aux1,'');
  inicpila(aux2,'');
  if not filavacia(inicial) then
    apilar(aux1,extraer(inicial));
  while not filavacia(inicial) do
    if (primero(inicial)<=tope(aux1)) then
      apilar(aux2, extraer(inicial))
    else
      begin
        apilar(aux2, desapilar(aux1));
        apilar(aux1, extraer(inicial));
      end;
    while not pilavacia(aux1) do
      agregar(ultimo, desapilar(aux1));
    while not pilavacia(aux2) do
      agregar(inicial, desapilar(aux2));
  end;

begin
  inicfila(mayor,'');
  readfila(dada);
  while not filavacia(dada) do
    Sacamayor(dada,mayor);
  while not filavacia(mayor) do
    agregar(dada, extraer(mayor));
  writefila(dada);
end.

/*****/
```

```
Program E5P4;
{Separa la secuencia mas larga}
uses estructu;
var origen, secmaslarga:fila;
    ceros, mayor, mayor2, descarte:pila;

Procedure Sacaceros (var inicial:fila;var ultimo:pila);
{saca los ceros de inicial y los envia a ultimo}
begin
    while (not filavacia(inicial)) and (primero(inicial)=0) do
        apilar(ultimo, extraer(inicial));
end;

Procedure Sacasecuencias (var inicial:fila;var ultimo:pila);
{saca secuencias de numeros distintos de 0 de la fila inicial, y las
envia a ultimo}
begin
    while (not filavacia(inicial)) and (primero(inicial)<>0) do
        apilar(ultimo, extraer(inicial));
end;

Procedure Comparasecuencias (var
compara1,compara2,ceroscompara,eliminar:pila);
{compara las secuencias de compara1 y compara2, la mas larga la deja en
compara1, y
envia la menor a eliminar, luego desapila uno de ceroscompara, y lo apila
en eliminar}
var aux1,aux2:pila;
begin
    inicpila(aux1, '');
    inicpila(aux2, '');
while (not pilavacia(compara1)) and (not pilavacia(compara2)) do
    begin
        apilar(aux1, desapilar(compara1));
        apilar(aux2, desapilar(compara2));
    end;
if pilavacia(compara2) then
    begin
        while (not pilavacia(aux1)) and (not pilavacia(aux2)) do
        begin
            apilar(compara1, desapilar(aux1));
            apilar(eliminar, desapilar(aux2));
        end;
        apilar(eliminar, desapilar(ceroscompara));
    end
else
    begin
        while not pilavacia(aux1) do
            apilar(eliminar, desapilar(aux1));
        while not pilavacia(aux2) do
            apilar(compara2, desapilar(aux2));
```

```
    while not pilavacia (compara2) do
        apilar (compara1, desapilar (compara2));
        apilar (eliminar, desapilar (ceroscompara));
    end;
end;

begin
    readfila (origen);
    inicfila (secmaslarga, '');
    inicpila (ceros, '');
    inicpila (mayor, '');
    inicpila (mayor2, '');
    inicpila (descarte, '');
    Sacaceros (origen, ceros);
    Sacasecuencias (origen, mayor);
    while not filavacia (origen) do
        begin
            Sacaceros (origen, ceros);
            Sacasecuencias (origen, mayor2);
            Comparasecuencias (mayor, mayor2, ceros, descarte);
        end;
    while not pilavacia (mayor) do
        agregar (secmaslarga, desapilar (mayor));
    writefila (secmaslarga);
end.
```