

CREATING A CONSOLE APPLICATION

Console applications are pure 32-bit Windows programs that run without a graphical interface. When a console application is started, Windows creates a text-mode console window through which the user can interact with the application. These applications typically don't require much user input.

All the information a console application needs can be provided through

- a) Command line parameters,
- b) The ***readln()*** procedure or
- c) Text files. (e.g. input.txt)

All output will be via ***write(); and writeln();***

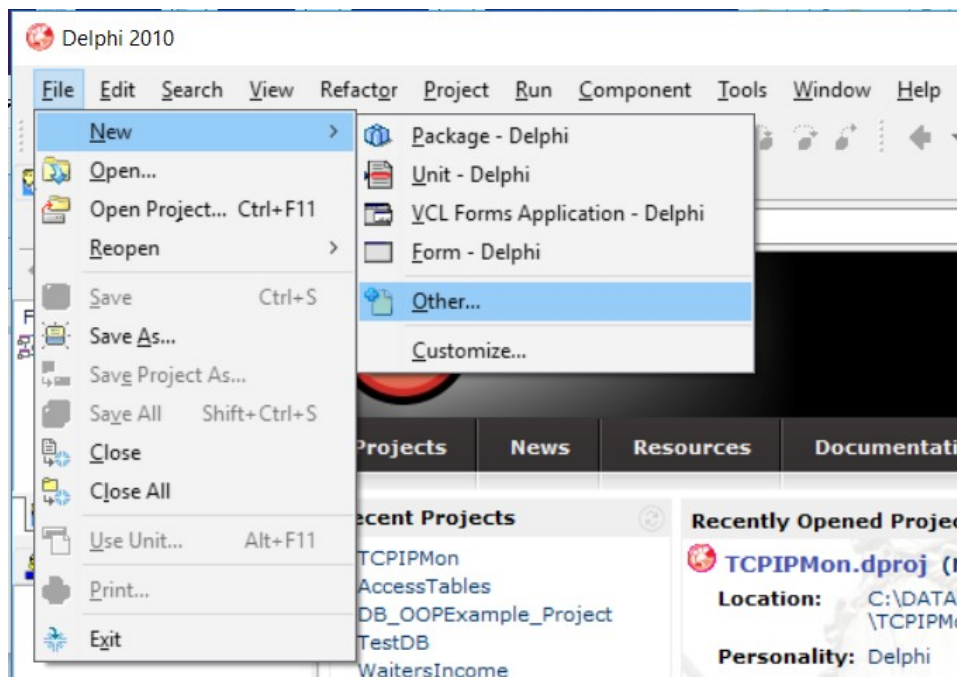
For new programmers, console applications simplify learning Pascal/Delphi - after all, the Pascal/Delphi introductory examples are nothing more than console applications.

STARTING A CONSOLE APPLICATION

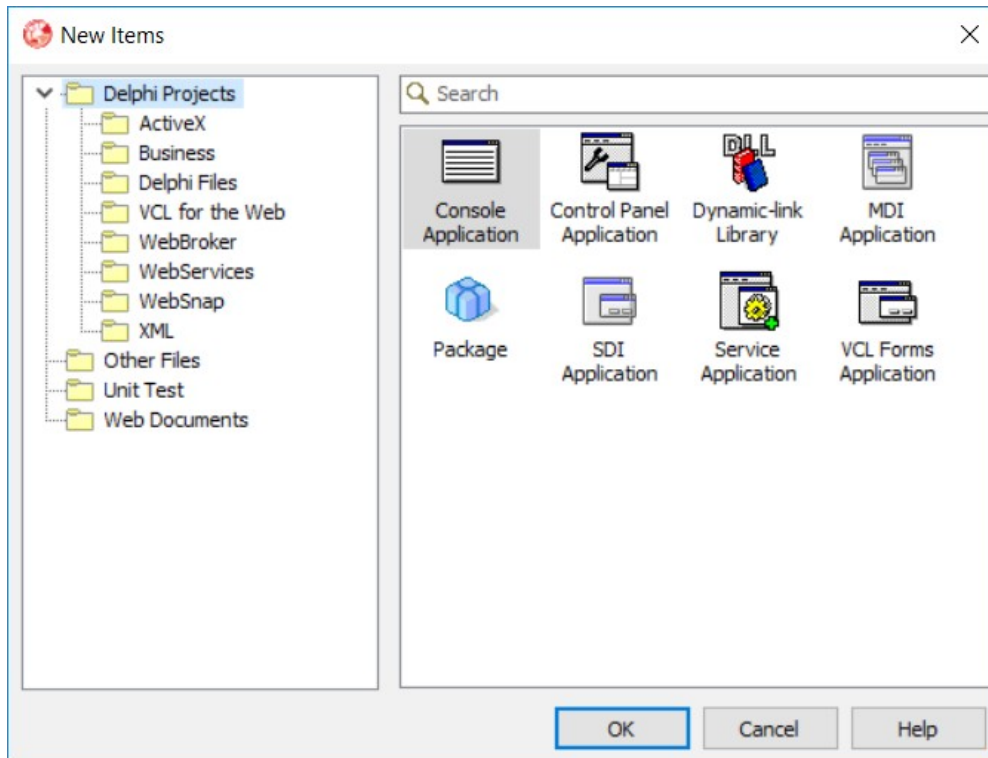
Here's how to quickly build console applications that run without a graphical interface. This assumes you are using the SA schools Delphi version 2010.

To create the application we use the Console Application Wizard by:

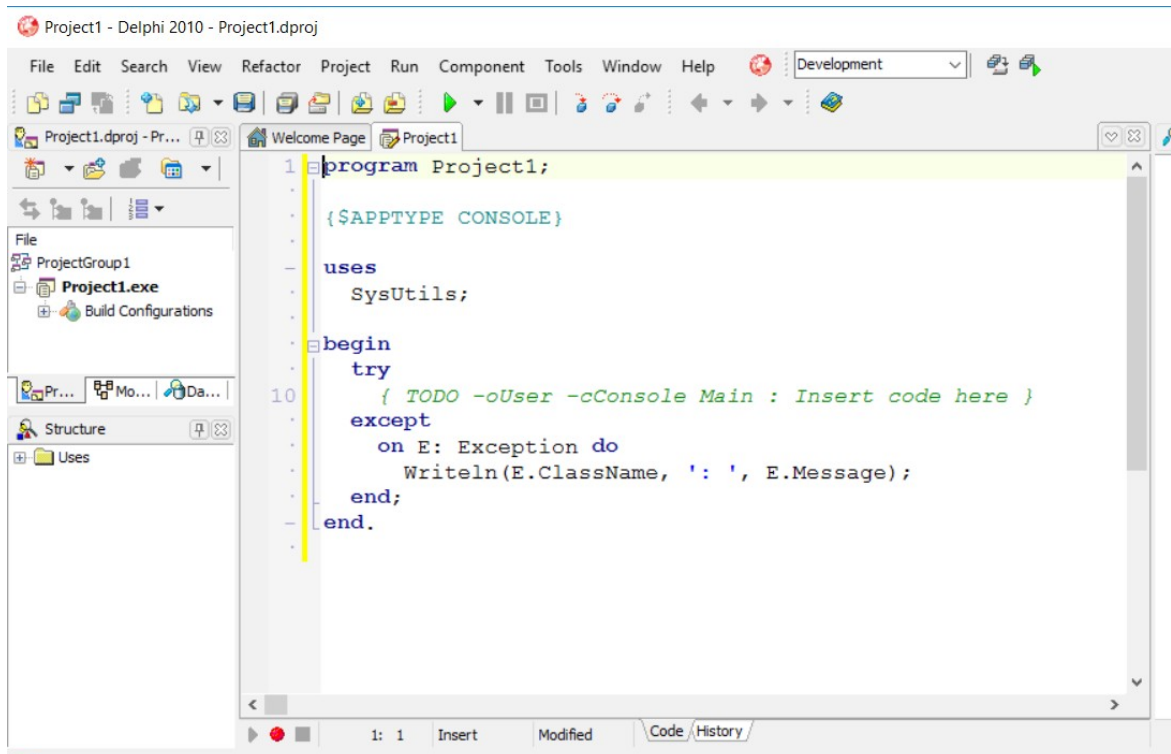
1. Going to **File** → **New**, this opens up a New Items dialogue, select **Other**



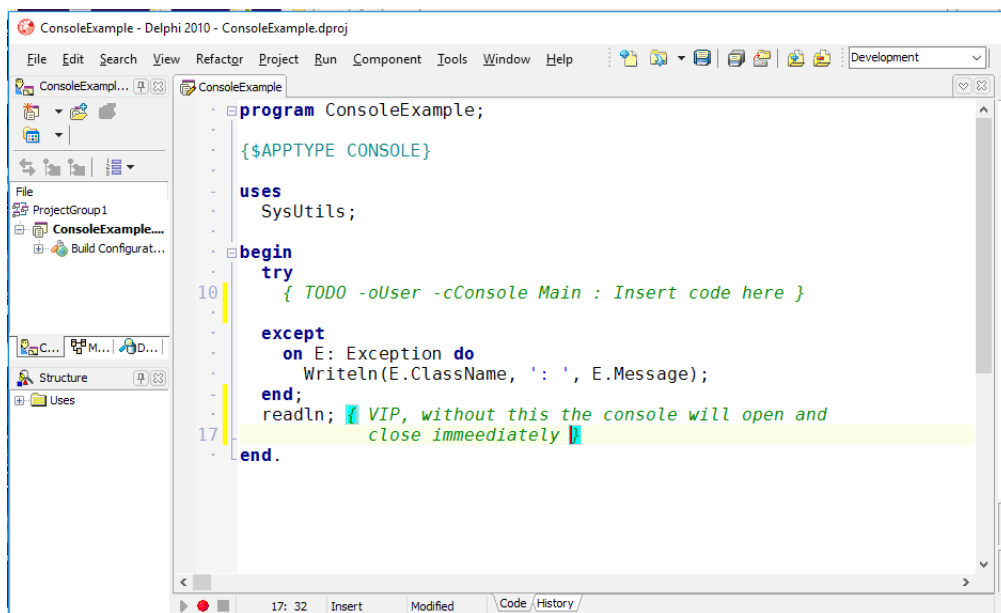
- In the **'New items'** window select the Console Application and press **OK** (or Double click the icon) and the wizard will set-up a Delphi project ready to be compiled as a console application.



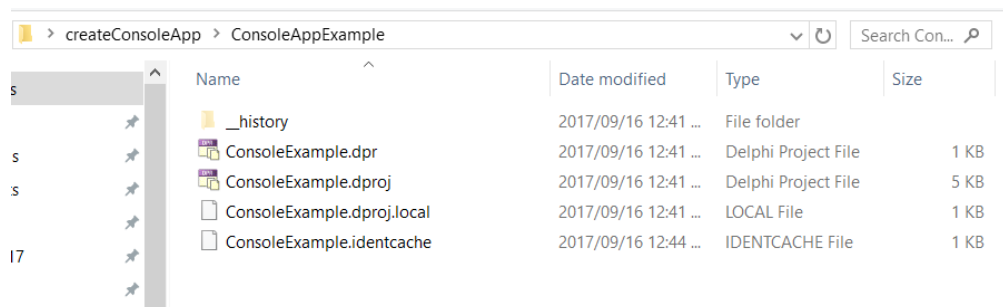
- The console Application you will be presented with, will look like this:



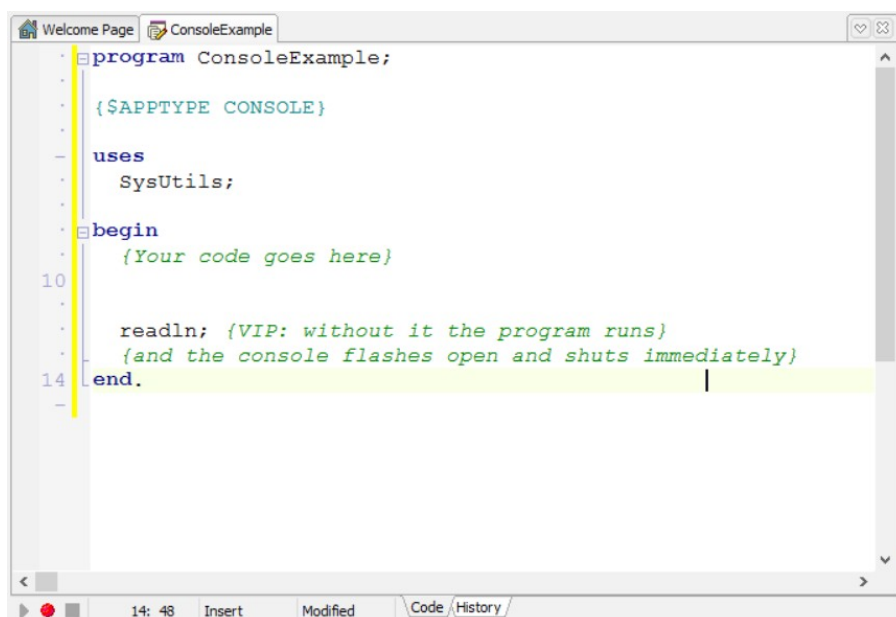
- Now select project, see highlight under ProjectGroup1, right click and select 'rename'



- Now add the **readln;** line 16 in the illustration above and save the project,
- To save press ctrl+shift+s, create the folder for your App and save the project in it. Note the console only has the **.dproj** and **.dpr** file (this holds the code). There is no **.pas** file.



- Another way to use the console is to set up a template as follows. Note the readln;



8. Here's how you would add the template (stub) supplied with the SACO question.

a) Go to the server <https://saco-evaluator.org.za/> follow the directions to the

Contest Server

Any currently running contests will be hosted on the Contest Server, as well as a selection of past contests intended for preparation purposes.

[Enter Contest Server](#)

For further help regarding the usage of the Contest Server, you can consult our [documentation manual](#). You can also watch a [tutorial video here](#).

b) press the button to enter

c) select a round

SAPO Contest Server

Choose a contest

- [SAPO 2019 Round 2](#)
- [SAPO 2019 Round 1](#)
- [SAPO 2018 Round 2](#)
- [SAPO 2018 Round 1](#)

d) 'Doubleclick' on the

PERFECT2

- [Statement](#)
- [Submissions](#)

PATTERN

- [Statement](#)
- [Submissions](#)

e) Doubleclick on the .pas 'Attachment'


[Statement in Afrikaans](#)


[Statement in English](#)


Some details

Type	Batch
Time limit	10.000 seconds
Memory limit	256 MiB

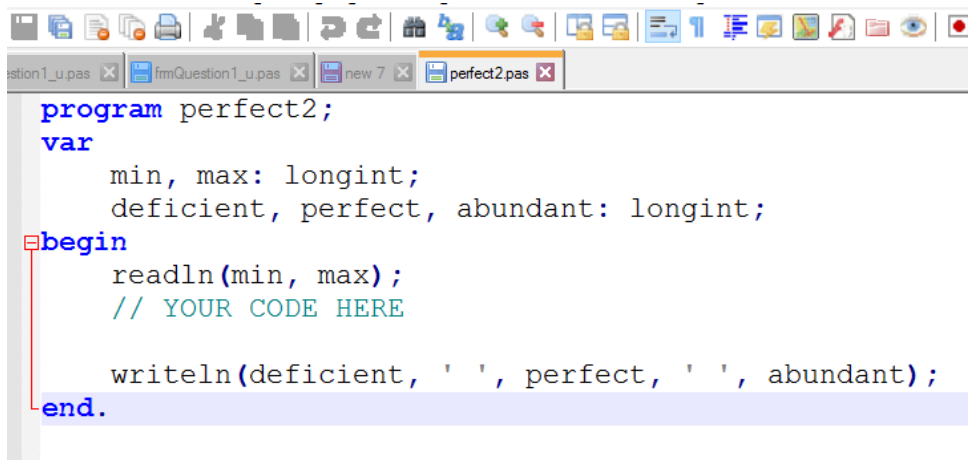
Attachments

 **perfect2.2.py** Python script 161 bytes

 **perfect2.cpp** C++ source code 216 bytes

 **perfect2.pas** Pascal source code 182 bytes

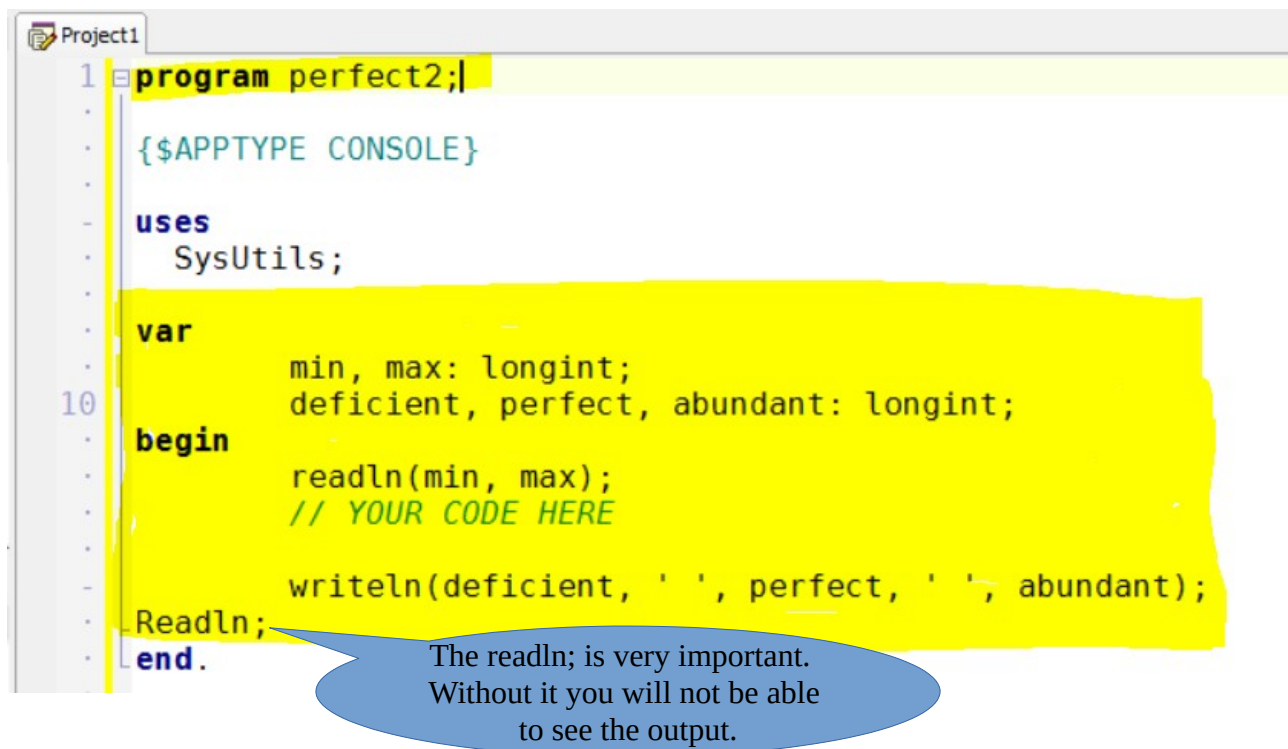
f) Open the .pas file in your favourite editor ...



```
program perfect2;
var
  min, max: longint;
  deficient, perfect, abundant: longint;
begin
  readln(min, max);
  // YOUR CODE HERE

  writeln(deficient, ' ', perfect, ' ', abundant);
end.
```

g) Paste the code into the Delphi console app as shown below.



```
1 program perfect2;
.
. {$APPTYPE CONSOLE}
.
. uses
.   SysUtils;
.
. var
.   min, max: longint;
10  deficient, perfect, abundant: longint;
.
. begin
.   readln(min, max);
.   // YOUR CODE HERE
.
.   writeln(deficient, ' ', perfect, ' ', abundant);
. Readln;
. end.
```

The readln; is very important. Without it you will not be able to see the output.

h) You are now good to go.

HAPPY PROBLEM SOLVING