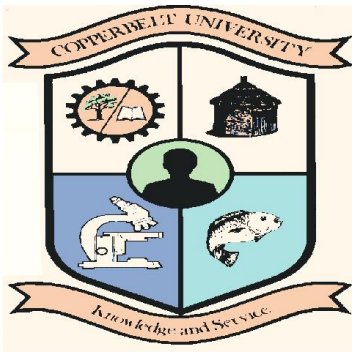

THE COPPERBELT UNIVERSITY

SCHOOL OF MATHEMATICS AND NATURAL SCIENCES

PE 231 PRACTICALS MENU

This Lab Menu is meant to help students' appreciate the basics of programming using Turbo pascal.

Compile by Dr Chaamwe

Table of Contents

1. Writing the First Program

The first thing to do is to open a text editor as shown in figure 1 below

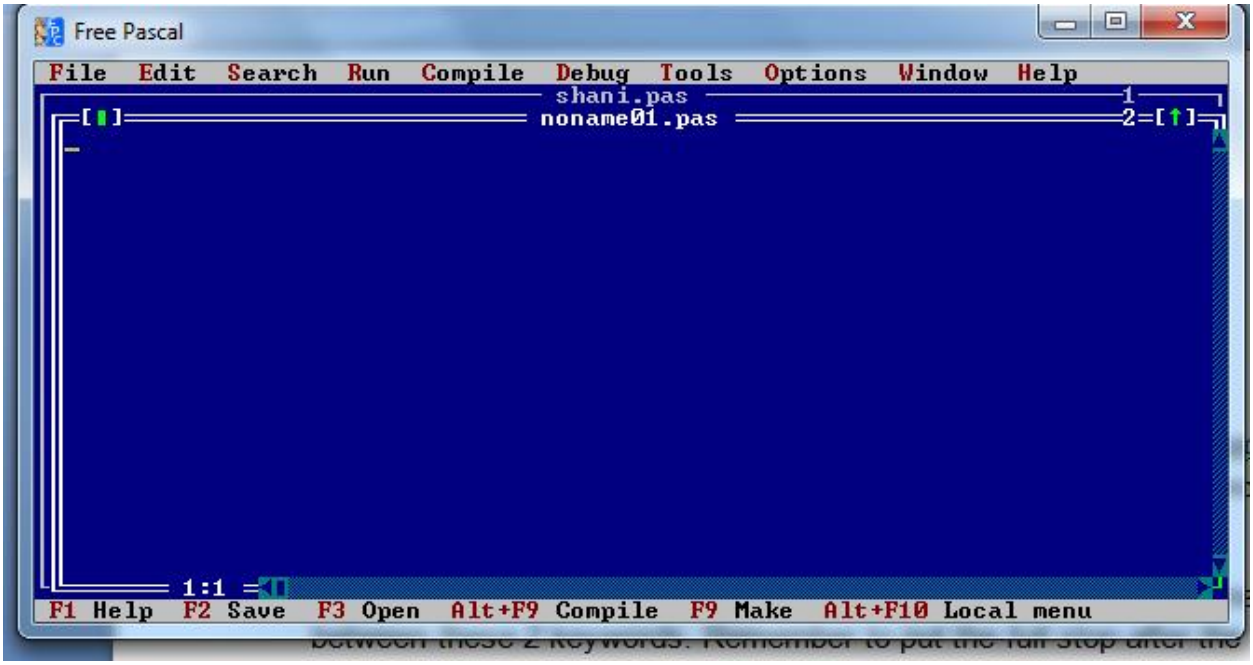

Figure 1: Pascal's Text editor

We always start a program by typing its name. Type *program* and the name of the program next to it. We will call our first program "Shani" because it is going to print the words "Mulishani Bonse" on the screen.

Next we will type *begin* and *end.* We are going to type the main body of the program between these 2 keywords. Remember to put the full stop after the *end*. Then write the command *Write* to print words on the screen.

```
program Hello;
begin
        Write('Mulishani Bonse');
end.
```

The *Readln* command will now be used to wait for the user to press enter before ending the program.

```
program Shani;
begin
        Write('Mulishani Bonse');
        Readln;
end.
```

You must now save your program as Shani.pas.

To compile the programme click on Compile on the main menu, or enter the following: fpc shani.pas

To run the program, click on run and you should see the words "Mulishani bonse" and pressing enter will exit the program.

2. More commands

Writeln is just like *Write* except that it moves the cursor onto the next line after it has printed the words. Here is a program that will print "Mulishani" and then "bonse" on the next line:

```
program Shani;
begin
        Writeln('Mulishani')
        Write("Bonse');
        Readln;
end.
```

If you want to skip a line then just use *Writeln* by itself without any brackets.

3. Using commands from units

The commands that are built into your Pascal compiler are very basic and we will need a few more. The crt unit is one of the most useful. The *ClrScr* command in the crt unit clears the screen. Here is how you use it:

```
program Shani;
uses
        crt;
begin
        ClrScr;
        Write('Mulishani Bonse');
        Readln;
end.
```

4. Comments

You should always have a comment at the top of your program to say what it does as well as comments for any code that is difficult to understand. Comments must be put between curly brackets. Here is an example of how to comment the program we just made:

```
{This program will clear the screen, print "Mulishani Bonse" and wait for the user to press enter.}
program Shani;
uses crt;
begin
        ClrScr;{Clears the screen}
        Write(Mulishani Bonse');{Prints "Mulishani Bonse"}
        Readln;{Waits for the user to press enter}
end.
```

5. Declaring and using Variables and Constants in Pascal

You must always declare a variable before you use it. We use the *var* statement to do this. You must also choose what type of variable it is. Here is a table of the different variable types:

Byte	*0 to 255*
Word	*0 to 65535*
ShortInt	*-128 to 127*
Integer	*-32768 to 32767*
LongInt	*-4228250000 to 4228249000*
Real	*floating point values*
Char	*1 character*
String	*up to 255 characters*
Boolean	*true or false*

You can create 2 or more variables of the same type if you separate their names with commas. You can also create variables of a different type without the need for another *var* statemtent.

```
program Variables;
        var i, j: Integer;
        s: String;
begin
end.
```

When you assign a value to a string variable, you must put it between single quotes. Boolean variables can only be assigned the values True and False.

```
program Variables;
 var
        i: Integer;
        s: String;
        b: Boolean;
begin
        i := -3;
        s := 'Hello';
        b := True;
end.
```

6. Calculations with variables

Variables can be used in calculations. Here is a table of the operators that can be used:

+	*Add*
-	*Subtract*
***	*Multiply*
/	*Floating Point Divide*
div	*Integer Divide*
mod	*Remainder of Integer Division*

The following example shows a few calculations that can be done:

```
program Variables;
var Num1,
        Num2,
        Ans: Integer;
begin
        Ans := 1 + 1;
        Num1 := 5;
        Ans := Num1 + 3;
        Num2 := 2;
        Ans := Num1 - Num2;
        Ans := Ans * Num1;
end.
```

Printing variables on the screen is just as easy. If you want to print variables and text with the same *Writeln* then seperate them with commas.

```
program Variables;
var
        i: Integer;
        s: String;
 begin
         i := 24;
        s := 'Shani';
        Writeln(i);
        Writeln(s,' Bonse');
end.
```

7. Constants

Constants are like variables except that their values can't change. You assign a value to a constant when you create it. *const* is used instead of *var* when declaring a constant. Constants are used for values that do not change such as the value of pi.

```
program Variables;
const
pi: Real = 3.14;
var
c, d: Real;
begin
d := 5;
c := pi * d;
end.
```

8. Decisions

8.1if then else

The *if* statement allows a program to make a decision based on a condition. The following example asks the user to enter a number and tells you if the number is greater than 5:

```
program Decisions;
var i: Integer;
begin
Writeln('Enter a number');
Readln(i);
if i > 5 then
Writeln('Greater than 5');
end.
```

>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
=	Equal to
<>	Not equal to

The above example only tells you if the number is greater than 5. If you want it to tell you that it is not greater than 5 then we use *else*. When you use else you must not put a semi-colon on the end of the command before it.

```
program Decisions;
var i:
Integer;
 begin
    Writeln('Enter a number');
   Readln(i);
  if i > 5 then
      Writeln('Greater than 5')
  Else
     Writeln('Not greater than 5');
end.
```

If you want to use more than 1 condition then you must put each condition in brackets. To join the conditions you can use either *AND* or *OR*. If you use *AND* then both conditions must be true but if you use *OR* then only 1 or both of the conditions must be true.

```
program Decisions;
var
i: Integer;
begin
   Writeln('Enter a number');
    Readln(i);
     if (i > 1) and (i < 100) then
        Writeln('The number is between 1 and 100');
end.
```

If you want to put 2 or more commands for an *if* statement for both the then and the else parts you must use *begin* and *end;* to group them together. You will see that this *end* has a semi-colon after it instead of a full stop.

```
program Decisions;
var i: Integer;
begin
Writeln('Enter a number');
Readln(i);
if i > 0 then
        begin
        Writeln('You entered ',i);
        Writeln('It is a positive number');
        end;
end.
```

You can also use *if* statements inside other *if* statements.

```
program Decisions;
var
i: Integer;
begin
   Writeln('Enter a number');
   Readln(i);
  if i > 0 then
     Writeln('Positive')
  else
    if i < 0 then
    Writeln('Negative')
    else
        Writeln('Zero');
end.
```

9. Loops

Loops are used when you want to repeat code a lot of times. For example, if you wanted to print "Shani" on the screen 10 times you would need 10 *Writeln* commands. You could do the same thing by putting 1 *Writeln* command inside a loop which repeats itself 10 times.

There are 3 types of loops which are the *for* loop, *while* loop and *repeat until* loop.

9.1For loop

The for loop uses a loop counter variable, which it adds 1 to each time, to loop from a first number to a last number.

```
program Loops;
 var i: Integer;
begin
for i := 1 to 10
          do Writeln('Hello');
 end.
```

If you want to have more than 1 command inside a loop then you must put them between a *begin* and an *end*.

```
program Loops;
var i: Integer;
begin
          for i := 1 to 10 do
          begin
            Writeln('Shani');
           Writeln('This is loop ',i);
        end;
 end.
```

9.2While loop

The *while* loop repeats while a condition is true. The condition is tested at the top of the loop and not at any time while the loop is running as the name suggests. A while loop does not need a loop variable but if you want to use one then you must initialize its value before entering the loop.

```
program Loops;
var
i: Integer;
begin
    i := 0;
  while i <= 10
     begin i := i + 1;
        Writeln('Shani');
    end;
end.
```

9.3Repeat until loop

The *repeat until* loop is like the *while* loop except that it tests the condition at the bottom of the loop. It also doesn't have to have a *begin* and an *end* if it has more than one command inside it.

```
program Loops;
var
 i: Integer;
begin
     i := 0;
    repeat i := i + 1;
        Writeln('Hello');
    until i = 10;
end.
```

If you want to use more than one condition for either the *while* or *repeat* loops then you have to put the conditions between brackets.

```
program Loops;
 var
i: Integer;
s: String;
begin
    i := 0;
    repeat i := i + 1;
       Write('Enter a number: ');
      Readln(s);
   until
   (i = 10) or (s = 0);
 end.
```

LAB EXERCISES

1. Write a pascal program that calculates your age.
2. Write a pascal program that calculates your final exam mark given the following distribution of marks: CA: 40 (test1: 15, Test 2: 15 and Assignment: 10) and Exam 60 (marked out of 100).
3. Write a pascal program which inputs student marks, and prints the grades as follows: below 40, D, between 40 and 50, D+, between 50 and 65, C, between 65 and 75, B and above 75, A
4. Write a pascal program that calculates the total surface area of a cone.
5. Write a pascal program that calculates the total surface area of a right cylinder.