UNIVERSITEIT•STELLENBOSCH•UNIVERSITY
jou kennisvennoot • your knowledge partner

Computer Programming 143

Practical 2

2019

Aim of Practical 2:

1. Code **while** loops.
2. Code **for** loops.

For some tongue-in-cheek flowcharts visit **xkcd**, *the links are:* xkcd: Flow Charts *and* xkcd: Good Code

"`while(1);` *//So that is what infinity looks like.*"

Instructions

1. Attendance is **compulsory** for all the practical sessions of your assigned group. See the study guide for more details.
2. The last section (usually the last 30 minutes) of the practical will be used for a test.
3. If more than two tests have been missed for what ever reason, you will receive an **incomplete** for the subject. See the study guide for more details.
4. You must do all assignments **on your own**. Students are encouraged to help each other **understand** the problems and solutions, but each should write his/her own code. By simply copying someone else's code or solutions, you will not build an understanding of the work.
5. You are responsible for your own progress. Ensure that you understand the practical work. Check your work against the memorandum that will be posted on Wednesday afternoons on learn.sun.ac.za.
6. Use H:\CP143 as your Code::Blocks workspace folder for all projects. But it is highly suggested that you also use a **flash drive to backup** all your work.
7. Create a new project **for each assignment**. See *Creating a Code::Blocks Project* in Practical 0 for instructions on how to do this.
8. Include a comment block at the top of each source file according to the format given. It must include the correct filename and date, your name and student number, the copying declaration, and the title of the source file.
9. **Indent your code correctly.** Making your code readable is not beautification, it is a time and life saving habit. Adhere to the standards (refer to the documents on SUNLearn).
10. Comment your code sufficiently well. It is required for you and others to understand what you have done.

Assignment A

Goal:

Use ***`while`*** *and* ***`if`*** *statements to write a simple menu which gives the user a number of choices.*

Getting started

1. Create a project named `Assignment2A`. Make sure that this is the only open project in the workspace before compiling the program.

2. Include the **standard comment block** above your main function. Also, comment your whole program appropriately.

3. Read the complete question (point 1 to point 6) before you start programming.

4. Either draw a program flowchart or write the pseudocode **before** you start programming. A demi/instructor may ask to see this when they come to assist you.

5. Below are two sections: a **description of what your program should do**, and some **guidelines for how to write the program**. If you think you know how to write the program after just reading the program description, you can start programming. Otherwise, read the programming guidelines (after the sample outputs) before starting.

Program description

1. Print a menu with three options for the user: the first to do some circle calculations, the second to do some triangle calculations and the third to exit the program. See point 6 on the next page for an example on how the output should look.

2. Allow the user to enter their choice. Use **`if`** statements to execute the operation of their choice. Display an error message if the choice is not valid.

3. After executing the user's choice (or displaying an error message), display the menu again. This sequence of displaying the menu, waiting for user input and then executing commands according to that input should **repeat** till the user enters the *Exit Program* option. You should implement this using a **`while`** loop.

4. If the user's choice was circle calculations, ask for the radius of a circle and read it in as a **float**. Display the circumference and the area of the circle. Display only the first four digits after the decimal point for each. Use the value 3.1416 for π.

5. If the user's choice was triangle calculations, ask for three **integers**. Determine if they could be the sides of a right triangle (remember Pythagoras!), with the **first side entered** being the hypotenuse (skuinssy). Print a statement to display your finding.

6. Sample run of the program:

```
Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 1
Please enter the radius of the circle: 7.5
The circumference is 47.1240 and the area is 176.7150.

Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 2
Enter hypotenuse: 5
Enter side A: 4
Enter side B: 3
The first number is the hypotenuse of a right triangle.

Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 2
Enter hypotenuse: 3
Enter side A: 4
Enter side B: 5
The first number is NOT the hypotenuse of a right triangle.

Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 2
Enter hypotenuse: 9
Enter side A: 7
Enter side B: 8
The first number is NOT the hypotenuse of a right triangle.
```

```
Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 4
An invalid menu option.

Menu Options:
1. Circle Calculations
2. Triangle Calculations
3. Exit Program

Enter your choice: 3
Quitting...
```

Programming guidelines

1. Start by writing code that prints the menu on screen and asks the user for his choice.

2. Either add an `if..else` statement which checks what the user entered, and prints a message on the screen saying what the user chose (you will delete the `printf` statement printing this message later); or alternatively, use the debugger with a breakpoint to check the user choice (after the `scanf` instruction). Watch this video to learn how to use breakpoints: https://www.youtube.com/watch?v=IN_RTt_5cf0

   Now your program output should look like this, if for example the user chose option 1:

   ```
   Menu Options:
   1. Circle Calculations
   2. Triangle Calculations
   3. Exit Program

   Enter your choice: 1
   User chose circle calculations
   ```

3. If you have used extra `printf` statements, then you can delete them under each option in the `if..else` else statement. Update your program to perform the necessary calculations and to display the results.

4. Now enclose the code that prints the menu and the `if..else` statement in a `while` loop, so that the program will keep on executing until the user enters a 3. **Important:** Make sure that the program first prints the menu, and then checks if it should exit.

5. Ensure that you copy the **Assignment2A** project folder to a flash drive as a backup.

Assignment B

Goal:

Use `for` loops to print figures.

Getting started

1. Create a project named `Assignment2B`. Make sure that this is the only open project in the workspace before compiling the program.

2. Include the **standard comment block** above your main function. Also, comment your whole program appropriately.

3. Read the complete question before you start programming.

4. Either draw a program flowchart or write the pseudocode **before** you start programming. A demi/lecturer may ask to see this when they come to assist you.

Program description

1. The following program displays a square of 5 x 5 stars. If you would like to change the size of the grid you would have to change the code or write a new program or function. This is known as hard coding. A far better way would be to print it **dynamically**. Besides, what will happen if you want a grid of 100 x 100? You will have to count very carefully to put one hundred stars in one `printf` (to make the columns) and then copy that `printf` 100 times (to make the rows). Using `for` loops to do this job for you makes much more sense.

```c
#include <stdio.h>

int main(void)
{
        //printing a grid of size 5 x 5
        printf("\n* * * * *");
        printf("\n* * * * *");
        printf("\n* * * * *");
        printf("\n* * * * *");
        printf("\n* * * * *");
        return 0;
}
```

2. Write a program that reads in a grid size from the user. Print a grid of stars accordingly. To be able to do this, you will need two for loops, one **inside** of the other. The outer `for` loop will be for the rows, the inner `for` loop will print the columns. Only one star and a space will be printed at a time.

3. Sample output of the program:
   Sample 1:
   ```
   Enter the grid size: 2

   * *
   * *
   ```
   Sample 2:
   ```
   Enter the grid size: 13

   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   * * * * * * * * * * * * *
   ```

Programming guidelines

1. First change the given program so that it contains a single `printf` statement, which prints five stars. This `printf` should be inside a **`for`** loop, so the statement will be executed five times to print five rows of stars.

2. Now change the `printf` statement to print only one star, and put it inside a second **`for`** loop, *inside* the first one, so that five stars will be printed next to each other during each iteration of the outer **`for`** loop. Make sure that your **`for`** loops have curly brackets where necessary, and that newlines are printed in the correct place in your code.

3. Finally, change your code so that it does not print a 5 x 5 grid every time, but asks the user for a grid size and then prints a grid of that size.

4. Ensure that you copy the **Assignment2B** project folder to a flash drive as a backup.

Something to think about: Exercise 4.16 on page 167 in the prescribed textbook is a beautiful example of really getting to grips with **`for`** loops. Give it a go.

Assignment C

Goal:

Practise writing pseudocode and drawing flow diagrams

1. A program does the following: Display a menu that will give the user five choices of operations to perform. Read their choice (display an error message if their choice is invalid). Print a message asking for the first number and read it in. Do the same for the second number. Execute the chosen operation on the two numbers. Do NOT include error checking, such as checking for a non-zero value. Print the result of the calculation. Use the sample output in the next point as a guideline.
2. Write the pseudocode for this program. You can do this on paper or in Notepad or Word or any other editor.
3. Draw a flow chart for the same program.
4. Sample output:

```
Menu of Operations
------------------
1.  +
2.  -
3.  *
4.  /
5.  %
Enter operation number to be executed: 1
Enter first integer: 12
Enter second integer: 34

12 + 34 = 46
```