Statement to be included in assessments and assignments (as approved by the Registrar, Vice-Principal Academic and DESA): *"The University of Pretoria commits itself to produce academic work of integrity. I affirm that I am aware of and have read the Rules and Policies of the University, more specifically the Disciplinary Procedure and the Tests and Examinations Rules, which prohibit any unethical, dishonest or improper conduct during tests, assignments, examinations and/or any other forms of assessment. I am aware that no student or any other person may assist or attempt to assist another student, or obtain help, or attempt to obtain help from another student or any other person during tests, assessments, assignments, examinations and/or any other forms of assessment."*

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

COS341 Compiler Design
EO2: **FINAL EXAM**

Department of Computer Science

University of Pretoria

July 2020: **3-hours work-time, from home**

Internal Examiner: Prof. S. **Gruner**.
External Examiner: Prof. B. **Watson (*Stellenbosch*)**

Instructions!

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- **Solve all problems** with help of your **textbook!**
- The Total Value of this Exam is ***40 Points***:

  ***40 Points* = 100% Percent**.
- Provide your solutions **as PDF**:
  - Word.DOC files can<u>not</u> be accepted under any circumstances!
- **Upload** your solutions to the course web page:
  - EMail submissions can<u>not</u> be accepted under any circumstances!
- Submit your solutions **timely before deadline**:
  - Belated submissions can<u>not</u> be accepted under any circumstances!

Time Advice!

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- The 3 hours (**180 minutes**) available for this exam should be spent approximately as follows:
  - approximately **70 minutes** for **Task 1**
  - approximately **80 minutes** for **Task 2**
  - approximately **30 minutes** for **Task 3**

IMPORTANT !
During the running exam, make **frequent safety-uploads** of your semi-finished solutions **!** In the case of any technical problems during the exam, for example a temporary internet outage, you'll have at least the safety-uploads which can be marked. Also make sure that your laptop's battery is fully charged, in case of a temporary electricity outage **!**

Chapter Advice:

- The **three main Tasks** of this Exam are related to the ***Study-Scope*** as follows:
  - **Task 1 ←→** *mainly to Chapter* ***#6***,
  - **Task 2 ←→** *mainly to Chapter* ***#8***,
  - **Task 3 ←→** *mainly to Chapter* ***#9***.

Preliminary Remarks

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- **Throughout all the following Tasks**,
  - **all variables are assumed to be integer** variables, such that we can omit any explicit "int" declaration statements (for convenience);
  - the Boolean **truth values will be represented by integers** as well {0,1}, such that we an omit any explicit "bool" declaration statements;
  - capital "**N**" represents some **constant Number** (integer) **which is somehow provided by the user**; in other words: "**N**" is *not* a variable!

The Collatz Program

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- The basis of all your subsequent to-do Tasks is the **Collatz** program, the code of which is presented on the following slides.
  - Familiarise yourself with this program before you proceed to the to-do-tasks!
  - Note that the given Collatz program has five variables: **i**, **o**, **e**, **t**, **c**.
- The Task specifications appear on the slides “after” the program code.

The Collatz Program


```
i  := N; // comment: input Number from user
o := 0; // comment: o is the output variable
e := 0; // comment: e is a "pseudo-boolean", needed for an "even number" test
t  := 0; // comment: t is an auxiliary variable needed for the "even number" test
c := 0; // comment: c is counting the iterations of the outer loop
WHILE ( i > 1 ) DO
    {
     t := i;
     WHILE ( t > 1 ) DO // comment: now we test for "even number"!
         {
          t := t – 2;
          IF ( t == 0 ) THEN { e := 1 } // comment: true, even number!
         }
     IF ( e == 1 ) THEN { i := i / 2 } ELSE { i := 1 + ( 3 * i ) }
     e := 0; // comment: re-set the pseudo-boolean for the next even-test!
     c := c + 1; // comment: another outer loop is done: counter goes up.
    }
o := o + c; // comment: end of program: o shows the output result
```

Additional Comments

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- **For "*educational purposes*" the given formulation of the Collatz program is indeed slightly *sub-optimal*:**
  - Variable **c** could be eliminated altogether by
    - replacing "**c := c + 1**" in the loop with "**o := o + 1**",
    - which would also allow us to delete the program's final line, "**o := o + c**"

Nonetheless we will work with the given sub-optimal version, because later we want to find out whether **c** and **o** "interfere" with each other...

Your TASKS →→→

Note:

Please upload a **separate solution document for each Task** in a separate **PDF** file (including your name and student-ID).

Do **NOT** merge all your Task solutions into one huge file with incredibly many megabytes**!**

Where you insert “photos” of “hand-drawn” pictures, please apply **low image-resolution**, such that the resulting upload-files do not become too large in megabytes**!**

Task 1 [*20 Points*]

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- **With help of the recursive translator function of Chapter #6, translate the Collatz program correctly "down" to intermediate code.**
  - **Important: Demonstrate and explain the "working" of the translation-process** as it happens from step to step, and **do NOT merely show the resulting intermediate code alone**!

→

Additional Advice to Task 1

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- For the sake of simplicity, please **KEEP the original variable names** (**i**, **o**, **e**, **t**, **c**) **in the generated intermediate code**, so that you do NOT need to re-name them to v0, v1, v2, (etc).
  - In other words: **you do NOT need the Symbol Table** from Chapter #3 !
- **Instead** of the textbook's newly generated internal variable names t0, t1, t2 (etc.), which are emitted by *newvar*(), use the internal new names **V1**, **V2**, **V3**, (etc.)
- For the Jump-Labels emitted by *newlabel*(), use **L1**, **L2**, **L3**, (etc.)

Task 2 [*14 Points*]

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- With the techniques of **Chapter #8**, subject your intermediate code (from Task 1) to the following **"liveness" analyses**:
  - For each generated code line number ***k***, compute the **variable sets**
    - **gen[*k*]**, • **kill[*k*]**, • **succ[*k*]**, • **in[*k*]**, • **out[*k*]**, • **interference[*k*]**
  - Draw a *picture* of the resulting ***interference graph***

- **Important**: Show and explain stepwise how you proceed to your solutions, and **do NOT merely display the final results** alone!

Task 3 [*6 Points*]

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- The “even-test” in the Collatz program could alternatively be implemented by a function call **e := CALL even(i)** ,
- whereby the function can be defined as:

```
even(x) // comment: function definition, whereby we assume that x ≥ 0
{
 t := x; // comment: t is now a LOCAL variable in the scope of even()
 IF ( t == 0 ) THEN { return(1) }              // comment: 1=True, t is even
 ELSE { IF ( t == 1 ) THEN { return(0) } // comment: 0=False, t is odd
            ELSE { t := t – 2;
                       r := CALL even(t) // comment: recursion, r is local var.
                       return(r) } }
}
```

➔

- On the basis of the explanations given in Chapter #9, **show in all relevant details the code which the compiler will generate for the given function <u>even</u>(x)**,
  - including:
    - The “**prologue**” code (for the run-time stack with local variables),
    - The “**body**” code (for the algorithm of the function itself),
    - The “**epilogue**” code (for the run-time stack with local variables).

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

- Also note:
  - It is up to you whether you want to present your solution in the “caller-saves” or in the “callee-saves” strategy:
    - Please indicate which strategy you have chosen.
  - Keep in mind that <u>even</u>(x) is *recursive*, which implies that its body-code must also contain code for a call to itself.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

END of Specification

There are no further Questions.

Double-check before submission

that your work shows **on each Task-Sheet**:

your **First Name** (given name)

your **Last Name** (family name)

your **Student ID** (number)