Technical University of Munich
School of Engineering and Design
Prof. Dr. Martin Werner

Exam Computational Foundations I (LRG0060, WS 2021/22)

Rules:

- These sheets stay closed until the official start of the examination is announced
- Already now put your name and Matrikel number onto the examination front page. Have your passport and student ID available on your desk
- No further items except this examination document, paper given out by us, and a few pens are allowed on your desk.
- If you have a question, raise your hand. If the answer is relevant to everyone, we will put it onto the whiteboard
- If you are done, please stay in your location quietly and double-check everything. If you need to go to the restrooms, raise your hands. Only one person is allowed to leave the room at the same time.

Full Name: _____

Matrikel No.: _____

Evaluation

Task 1: Multiple Choice	/ 5
Task 2: Algorithm Representation	/ 5
Task 3: Turtle Graphics	/ 5
Task 4: Bubble Sort	/ 15
Task 5: Turing Machine	/ 10
Task 6: Markov Algorithms	/ 10
Task 7: Understanding Recursion	/ 5
Task 8: The Stack Data Structure	/ 5
Task 9: Object Orientation	/ 5
Task 10: Composition and Recursive Data Structures	/ 5
Sum	/ 70

Task 1: Multiple Choice

(5 points.)

Answer the following questions with yes or no.
A wrong answer is counted as -1, a correct answer is counted as +1, an answer not given is counted as 0 (if you don't know the answer it is smarter not to give an answer!). If you reach more than 5 points, the points will become bonus points, if you reach less than 0 points, the result is 0 points.

Are the following statements true or false?		True	False
a	In MATLAB, subsets of data can be created with slicing.		
b	In MATLAB, one needs to allocate memory for an object before creating or using it using malloc and free.		
c	For arbitrary large n, an algorithm of runtime complexity $O(n)$ will become faster than an algorithm of $O(n^2)$.		
d	The SDL libraries are a collection of linear algebra routines such that we can easily solve numeric problems in C++.		
e	MATLAB is a compiled language generating native executables.		
f	C++ is a compiled language generating native executables.		
g	A complete binary tree with depth n has 2^n leaf nodes.		

Task 2: Algorithm Representation

(5 points.)

Consider the Algorithm in Listing 1

Listing 1: An imperative program

```
1    DECLARE A,B,C,D,E,RESULT
2    A = 1;
3    B = SQUARE(A);
4    C = 3;
5    D = 4;
6    E = C+D;
7    RESULT = E*B;
```

a. Write a table with a row for each line of the previous program and a column for each variable declared and give the value **after** the line has completed. Use a ? if the value is unknown (e.g., the variable has not yet been initialized). Include Line 1 (which just means every variable value is unknown).

b. Consider the following program:

```
1 A = CIRCLE(P, R);
2 B = CIRCLE(Q, R);
3 M1, M2 = INTERSECT(A,B);
4 L = LINE(M1,M2)
5 RESULT = INTERSECT(L, LINE(P,Q))
```

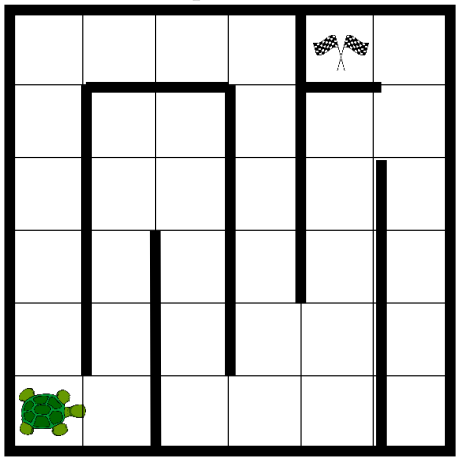
Give the same program as a single expression for result. That is, not using multiple lines of imperative code, but braces to organize the computation of the result.

c. Assume P, Q are given points, R is the radius of the circles and the length of the segment from P to Q, INTERSECT returns two intersection points of geometries (either of two circles or of two lines), LINE constructs a line between two points. Draw what the program does. Give variable names as labels for the objects you draw.

Task 3: Turtle Graphics

(5 points.)

Consider a turtle with commands `m` for move, `l` for turn left by 90° and `r` for turning right. The turtle starts in the marked position.



a. Sketch an imperative program that gets the turtle into the goal location not crossing bold walls. Denote the program as a string consisting of aforementioned characeters. For example `ml` means move and then turn left.

b. Now, we add a simple sensor to our turtle: It is always able to check whether it can move forward by running a C++ function `bool empty()`. In this context, the movement commands are implemented as C++ function `move()`, `turnleft()`, and `turnright()`. Give a short C++ program (without any function declaration, just the functional body) that solves the maze using `while` loops to avoid repetitions of commands. Write up to three columns - the program is not short.

Task 4: Bubble Sort

(15 points.)

Consider the C++ program in Listing 2. The function print just outputs a space-separated list of the values of the argument vector and its implementation details can be ignored.

Listing 2: Bubble Sort

```
1  #include<iostream>
2  #include<vector>
3  #include<iterator>
4  using namespace std;
5  void print(vector<int> const &input)
6  {
7      copy(input.begin(),input.end(),ostream_iterator<int>(cout, " "));
8  }
9
10 int main(void)
11 {
12    vector<int> A {9,2,6,3,7,4};
13    cout << "start; A=";  print(A); cout << endl;
14
15    for (size_t n=A.size(); n>1; n--)
16    {
17       for (size_t i=0; i<n-1; i++){
18           if (A[i] > A[i+1]){
19               auto tmp = A[i];
20               A[i] = A[i+1];
21               A[i+1] = tmp;
22           }
23      }
24      cout << "n: " << n << "; A=";
25      print(A);
26      cout << endl;
27   }
28 }
```

a. How often is the inner loop (L17-L22) executed?

b. Create a table with the state after execution of Line 21 by reporting n, i, and the entries of A. Give the first 8 lines of this table only.

c. Define the term “Loop Invariant” in one sentence. Only the first given sentence is taken into account.

Task 5: Turing Machine

(10 points.)

Given a Turing machine with a finite string of 0s and 1s on the tape, delimited in both directions with empties ϵ, implement an algorithm that detects an instance of a three consecutive ones. Implement the Turing machine with states `start` for the beginning, `found` for the case that a triple one was found, and `notfound` otherwise. The band can be in arbitrary states afterwards. We only consider the final state as relevant.

a. Give the machine as a transition table. The table can be incomplete. As agreed, the machine will halt if there is no applicable rule.

b. Which of the following properties do Turing machines have? Answer yes or no:

- they are deterministic:
- they are not determined:
- they are capable of computing the sum of numbers:

Task 6: Markov Algorithms

(10 points.)

Markov algorithms have been introduced to simplify the analysis and implementation of Semi Thue Systems. *Note: This task illustrates, that Markov algorithms are Turing complete in that we give an example of an encoding of a Turing machine as a Markov algorithm.*

Listing 3: Implementing a Turing machine as a Markov Algorithm

```
1 S0 -> R0
2 R0 -> 0R
3 R1 -> 1R
4 R  -> C
5 1C -> C0
6 0C -> 1X
7 0X -> X0
8 1X -> X1
9 X -> H
```

a. What is the key difference (1 sentence) between Semi Thue systems and Markov algorithms?

b. Execute the given Markov algorithm on the input strings “S001” and “S010”. Give all intermediate strings and (preferably for correction) the line number of the rule you applied. Tip: Write your transitions below each other as most rules don’t change the length. Mark the left hand side (e.g., underline), replace it, and fill in the rest.

c. This program is essentially a Turing machine implemented as a Markov algorithm. Can you see how this was constructed? If so, can you apply the transformation yourself? Encode the following operations of a Turing machine as a Markov replacement:

   – From state X, reading a 1 on the tape, write a 0, move right, and go into state Y.

   – From state X, reading a 0 on the tape, write a 1, move left, and go into state Z.

Task 7: Understanding Recursion

(5 points.)

Consider the following program

Listing 4: "Recursion"

```
1  #include<iostream>
2  int B(int v);
3
4  int A(int v)
5  {
6     if (v >8) return B(v+2);
7     return A(v*2);
8  }
9  int B(int v)
10 {
11    while (v>5) return B(v-5);
12    return v-2;
13 }
14
15 int main()
16 {
17    std::cout << "A(" << 8 << ") == " << A(8) << std::endl;
18    std::cout << "A(" << 12 << ") == " << A(12) << std::endl;
19 }
```

a. Give the output of the program

b. For the first invocation `A(8)` give the complete sequence of invocations including the parameter. For example "$A(4) \Rightarrow A(5) \Rightarrow B(2)$"

c. [2 Bonus Points] For which inputs does the function A terminate? Give a short reason. Ignore possible overflow or underflows from finite arithmetic representation.

Task 8: The Stack Data Structure

(5 points.)

In the lecture, we introduced the stack data structure with three operations.

a. Name these operations and describe each of them in a single (short!) sentence.

b. Recursion is tightly linked to the "call stack". What are the stack operations used for? Please just name the four categories of things that are put onto the stack during function execution.

Task 9: Object Orientation

(5 points.)

Shorty answer the following questions on object orientation.

a. Define what Encapsulation means (in one sentence, roughly)

b. What is the difference between a function and a `virtual` function?

c. Why doesn't the following program compile?

Listing 5: Object Orietntation

```
1  #include<iostream>
2  class A
3  {
4     public:
5        int v,w;
6  };
7  class B: public A
8  {
9     private:
10      int x;
11    public:
12      B(): x(0){std::cout << "B's default constructor called"<<std::endl
            ;};
13      B(int _v) {v = _v; std::cout << "B's (int) constructor called"<<std
            ::endl;};
14 };
15 int main()
16 {
17    B* instance = new B(42);
18    std::cout << instance->x << std::endl;
19    return 0;
20 }
```

d. What is the output if the error is corrected?

Task 10: Composition and Recursive Data Structures

(5 points.)

In the lecture, we have learnt about recursive data structures. Answer the following questions:

a. Give the recursive data structure "single linked list" in the form of a text replacement system.

b. Declare a class in C++ which could be used to implement a list. Give no methods except a constructor that ensures proper initialization of the needed pointer variable. (The answer will typically have less than 10 lines, we are not picky about syntax. if it is semantically sensible, you get all points!))