



**American University of Beirut**  
**Faculty of Arts and Science,**  
**Department of Mathematics and Computer Science**

**CMPS258 final exam**  
**Programming Languages**  
**Exam duration: 2 hours**

Name: \_\_\_\_\_

ID: \_\_\_\_\_

**Exam guidelines:**

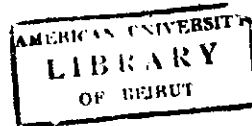
- a. students must submit this booklet as their answer sheet.
- b. students must use ink
- c. exam total marks is 51/50

**Part 1: (12 marks) Tick True (T) or False (F) each question is worth 0.5 mark.**

- |   |             |
|---|-------------|
| 1. Any object file contains a symbol table  | T ___ F ___ |
| 2. Lazy evaluation is essential to implement functions  | T ___ F ___ |
| 3. The linking process generates an executable file   | T ___ F ___ |
| 4. C++ references are easier to use than C pointers   | T ___ F ___ |
| 5. A symbol table in <i>code.o</i> has a list of the names of all variables declared in <i>code.c</i> | T ___ F ___ |
| 6. Not all object files become input to the linking process.  | T ___ F ___ |
| 7. The linking process builds a combined symbol table   | T ___ F ___ |
| 8. All pointers to void need to be cast before they can be used to reference an object                | T ___ F ___ |
| 9. The linking process searches library files for undefined symbols                                   | T ___ F ___ |
| 10. The default access modes for C++ structures and C++ classes are different                         | T ___ F ___ |
| 11. Currying is using partially implemented functions   | T ___ F ___ |
| 12. C++ references are used to pass parameters to functions   | T ___ F ___ |
| 13. The linking process checks for storage space  | T ___ F ___ |
| 14. 'Upcasting' is a mechanism underlying inheritance in any OO language                              | T ___ F ___ |
| 15. C++ structures cannot be used to implement inheritance  | T ___ F ___ |
| 16. C++ references are a useful aliasing mechanism  | T ___ F ___ |
| 17. Pointers to void are the same size as any other pointer   | T ___ F ___ |
| 18. Under some conditions an object file can be executed  | T ___ F ___ |
| 19. C++ references are generally easier to use than C pointers  | T ___ F ___ |
| 20. Pointers to void can be used as an information hiding mechanism                                   | T ___ F ___ |
| 21. C++ structures are used less often than C++ classes   | T ___ F ___ |
| 22. Type variables in Haskell implement polymorphism  | T ___ F ___ |
| 23. 'Downcasting' allows access of a base class through a pointer to a derived class                  | T ___ F ___ |
| 24. Any C++ reference requires four bytes of memory   | T ___ F ___ |

**Part 2: Short questions on C++ (7 marks) only one answer for a multiple choice question must be chosen. Each question is 1 mark**

- 1. C++ uses static binding for a function *f* of a derived class *D* if
  - *f* is declared virtual
  - *f* in the base class is virtual
  - *f* is virtual in both, *D* and the base class of *D*
  - *f* is not virtual anywhere



2. A function  $F$  in a base class  $B$  is always overridden by all derived classes. Therefore:
- $F$  should not be included in the interface of  $B$
  - $F$  should be declared as a virtual function in  $B$
  - Suggests that  $B$  is an abstract class
  - None of the above
3. Which of these is false about 'Downcasting'?
- Allows access of a derived class using a pointer to a base class.
  - Is not an implicit mechanism in C++
  - Is a default feature of pointers to objects in a class hierarchy in C++
  - None of the above
4. Consider the following C++ declarations:
- ```
int a=1, b=2;           /* line 1 */  
int &r_b, &r_a=a;       /* line 2 */  
int *a_ptr = &r_a;     /* line 3 */
```
- Lines 2 and 3 are faulty.
  - Line 2 is only faulty.
  - Line 3 is only faulty.
  - All lines are correct
5. If no access specifier is used, the default access mode of members in C++ classes is \_\_\_\_\_
6. Which C++ programming mechanism is not used to implement polymorphism:
- Inheritance
  - References
  - Function overloading
  - Operator overloading

7. Given the following inheritance relationships:

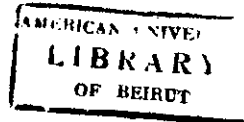
```
class A { ... };  
class B: public A { ... };  
class C: public B { ... };
```

Consider the object declarations:

```
A o1; /* line 1 */  
B o3; /* line 2 */  
C o5; /* line 3 */
```

Which statement is false:

- Line 1 causes the constructor of  $A$  to be called first.
- Line 2 causes the constructor of  $A$  to be called first.
- Line 3 causes the constructor of  $B$  to be called first
- None of the above



**Part 3: Short answers on Haskell (7 marks)**

1. Consider the following function: (1 mark)

```
my_length :: [a] → Int
my_length list
  | rest == []      = ???
  | otherwise      = 1 + my_length rest
  where
    rest = tail list
```

What is missing at ??? in my\_length to calculate the length of a non-empty list? \_\_\_\_\_

2. What is the type of each of the following expressions: (2 marks)

- a. `((5, 6, 'E', 'F')):[]` \_\_\_\_\_
- b. `(["a", q], ("b", 2), 'c')` \_\_\_\_\_

3. Consider these functions: (4 marks)

```
fib :: Int → Int
fib 0 = 1
fib 1 = 1
fib n = (fib (n-1)) + (fib (n-2))
```

```
sleep (first : rest) = first : (sleep rest)
sleep []             = []
```

```
even x = x 'mod' 2 == 0
```

What are the results of the following:

- a. `fib (-3)` \_\_\_\_\_
- b. `map fib [1..4]` \_\_\_\_\_
- c. `map sleep [[1..4], [1, 2]]` \_\_\_\_\_
- d. `filter even [1, 2, 3, 4, 5] ++ filter (not even) [4, 3, 2, 1]` \_\_\_\_\_

**Part 2 C++ programming (12 marks)**

For this question, use the space provided for your answers (and the back of your question sheets if needed). Consider the following C++ class interfaces that depict a simplified University structure. The university is organized as a collection of faculties. Each faculty is a collection of departments.

|                                                                                                                                                                                                                                                                                                          |                                                                                                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>class University { protected:     Faculty * faculties [10]; public:     Faculty * FindFaculty (char * faculty_name);     ~ University (); }</pre>                                                                                                                                                   | <pre>Class American_University : University { private:     int students_to_teachers = 25;     char * mission_motto; public:     American_University (int, char *); }</pre>             |
| <pre>class Faculty { private:     Department * departments [20];     char faculty_name [50];     int free_slot; // this contains the next free slot in departments     array public:     Department * FindDept (char * dept_name);     Department * AddDept (char * dept_name);     ~ Faculty(); }</pre> | <pre>class Department { public:     char department_name[50];     int total_majors; }  /* declaration of a university */  American_University: AUB(20, "We grow as we serve\n");</pre> |

Given the faculty name, *FindFaculty* returns a pointer to this faculty within an object *University*.

Given the department name, *FindDept* returns a pointer to this department within an object faculty.

Given a new department name, *AddDept* adds this department to the faculty (if possible) and it returns a pointer to this department in case it succeeds.

- a. Implement each of the three above methods (6 marks)



b. Implement the destructors of the classes *University* and *Faculty*. (2 marks)



c. Implement the constructor of the class *American\_University*. (2 marks)

d. Does the class *American\_University* require its own destructor? If yes, implement it; if not, why not? (2 marks)



**Part 3 Haskell programming (13 marks)**

**Q1. ( 6 marks)** Write a function which takes a string and gives you the *mirror* image of a string. A mirror image of a string is produced by reversing the string, and by also converting each character in the string as follows:  $a \rightarrow z$   
 $b \rightarrow y$   $c \rightarrow x$  .....  
e.g. *mirror* "abcd"  
"wxyz"

Your 'mirror' function should start here (Use the back of the sheet if you need more space) →



Q2. (7 marks) A Haskell function *interleave* interleaves three strings. The function type looks as follows:  
*interleave* :: *String* → *String* → *String* → *String*

Examples:

```
interleave "abcd" "1234" "ABCD"  
"a1Ab2Bc3Cd4D"  
interleave "abcdefg" "1234" "ABCD"  
"a1Ab2Bc3Cd4Defg"  
interleave "ab" "1234" "ABC"  
"a1Ab2B3C4"
```

The extent of the interleaving is determined by the length of the shortest string. Implement the *interleave* function.

*End of exam*