CS2104 Midterm AY2425S1

Question 1

Which of the following statements best describes the relationship between parsing and program execution?

- ☐ Parsing is part of the syntactic analysis of programs and is carried out before program execution.
- ☐ Parsing is part of lexical analysis, and its purpose is to provide program execution the strings that make up the program identifiers.
- ☐ Program execution and parsing usually overlap in time such that program execution can influence the result of parsing.
- ☐ Parsing is carried out after program execution and is used to format the result of execution.

Question 2

Which one of the following statements on interpreters is false?

- ☐ An interpreter is a program that specifies how other programs are executed.
- ☐ A hardware emulator is an interpreter where both source and target languages are machine languages.
- ☐ The purpose of an interpreter is to translate programs so that they can run on a platform that was not designed for the programming language that the program is written in.
- ☐ An interpreter usually incurs a runtime overhead when compared to running an equivalent program directly on a given hardware.

Question 3

Which one of the following statements on compilers is false?

- ☐ Compilers allow us to implement high-level programming languages with a performance that gets close to or even exceeds the performance of equivalent machine code programs.
- ☐ Compilers can be chained up so that programs can be compiled from high-level languages to lower and lower languages until we reach machine code.
- ☐ A compiler executes programs by "compiling" (translating) them to machine language.
- ☐ It is possible to compile a machine program into a high-level language such as Python.

Question 4

Which one of the following statements on the control of the CSE machine is true?

- ☐ The control of the CSE machine includes program fragments and control directives that are generated during program execution.
- ☐ The control of the CSE machine consists of a queue of statements and expressions all of which appear in the program that is being executed.
- ☐ The control of the CSE machine is a stack of directives all of which are generated while running the given program.
- ☐ The control of the CSE machine consists of a stack of statements and expressions all of which appear in the program that is being executed.

Question 5

Which one of the following statements on the stash of the CSE machine is true?

- ☐ The stash of the CSE machine will hold or point to the result of computation when program execution terminates.
- ☐ The stash of the CSE machine stores references to environments that need to be restored when functions return.
- ☐ The stash of the CSE machine aids in the program execution by mapping program names (identifiers) to their runtime values.
- ☐ The stash of the CSE machine includes program fragments that need to be executed in the future.

Question 6

Which one of the following statements on the environment component of the CSE machine is false?

- ☐ The runtime representation of functions (closures) store the environment from the time when the function was created.
- ☐ The control includes references to environments in special instructions so that the correct environment can be restored when functions return.
- ☐ When the control contains a name, the current environment is used to look up the name, possibly following a chain of environment frames.
- ☐ Environments store the arguments of primitive operations, when these arguments are primitive values such as numbers and boolean values.

Question 7

A function is recursive when it calls itself. Which of the following statements about recursion in the CSE machine is false?

- ☐ We can optimize recursion when the recursive call is the last instruction to be executed, because we do not need to return to the caller.
- ☐ The execution of recursive functions always requires a mark instruction to clear the control when a return instruction is encountered.
- ☐ Recursion usually involves the creation of a circular data structure in the environment.
- ☐ Without optimization, recursion will lead to a growth of the control stack that is at least proportional to the depth of the recursion.

Question 8

Which of the following statements on exception handling is true?

- ☐ Exceptions can be handled in the CSE machine similar to the handling of function calls.
- ☐ Exceptions can be handled in the CSE machine similar to the handling of function declarations..
- ☐ Exceptions can be handled in the CSE machine similar to the handling of conditionals.
- ☐ Exceptions can be handled in the CSE machine similar to the handling of blocks..

Question 9

Consider the following type declaration in SML:

```
datatype tree
    = Leaf of int
    | Node of {value: int,
              left:  tree,
              right: tree}
```

Which of the following statements is true?

- ☐ This type declaration is recursive, which is allowed in SML.
- ☐ This type declaration is not allowed because the type tree occurs in its own definition.

Question 10

Consider the following type declaration in SML:

```
datatype 'a stream
    = Nil
    | Cons of 'a * (unit -> 'a stream)
```

Which of the following statements is true?

- ☐ This recursive type declaration is not allowed in SML because the return type of the function in the second component of Cons is the datatype that is being defined.
- ☐ This recursive type declaration is possible in SML.

Question 11

Which of the following statements about memory safety is false?

- ☐ The language C relies on the underlying operating system to provide a rudimentary form of memory safety.
- ☐ A system is memory safe if all data is allocated statically.
- ☐ Type systems achieve memory safety by restricting memory access to operations that are explicitly designed by the programmer.
- ☐ Operating systems provide a form of memory safety by restricting memory access of each running process.

Question 12

Which of the following statements on dynamic typing is false?

- ☐ Dynamic typing allows programming language designers to avoid type declarations in programs.
- ☐ Dynamic typing ensures the safe execution of primitive operations, if necessary by terminating the running program to prevent unsafe operations at runtime.
- ☐ Dynamic typing usually incurs a runtime and/or memory overhead in the execution of programs.
- ☐ Dynamic typing prevents type errors by checking compliance with a type system before programs are executed.

Question 13

Which one of the following statements on type safety is the most accurate?

- ☐ Type safety is a property of entire programs.
- ☐ Type safety is a property of function declarations in programs.
- ☐ Type safety is a property of programming languages.
- ☐ Type safety is a property of primitive operations such as addition and accessor functions such as array access.

Question 14

Which of the following statements describes generic types most accurately?

- ☐ Generic types allow us to apply operations generically across a range of possible types.
- ☐ Generic types allow us to generate new data structures at runtime.
- ☐ Generic types are the principal mechanism for implementing inheritance in object-oriented programming languages.
- ☐ Generic types allow us to generate new functions at runtime.

Question 15

Which of the following statements about final declarations in Java is true?

- ☐ Final declarations in Java make sure that the data structures referred to by the final identifier are immutable.
- ☐ Final declarations in Java ensure that the program immediately terminates when a final value is reached.
- ☐ Final declarations in Java ensure that no other classes can access the identifiers that are declared final.
- ☐ Final declarations in Java prevent the declared identifier to be used as the target of assignment.

Question 16

JavaScript supports object access using an operation

```
object["key"]
```

Which one of the following statements about JavaScript's object access is correct?

- ☐ This operation plays a central role in JavaScript's support of exception handling.
- ☐ This operation plays a central role in JavaScript's support of object-oriented programming.
- ☐ This operation plays a central role in JavaScript's support of memory safety.
- ☐ This operation plays a central role in JavaScript's support of functional programming.