072-exam-2

Answer the following T/F questions, briefly explaining your answer. Turn in on paper.

Q. At runtime, every variable identifier has an associated area of memory which contains the variable's value.

Q. An L-value (left-hand side of an assignment expression) refers a memory location into which some value is written.

Q. An R-value (right-hand side of an assignment expression) refers to a memory location from which a value is read from memory, or refers to a CPU register from which a value is read.

Q. An array name is a pointer variable with its own associated memory, which at runtime contains the memory address of the memory location holding the first of the array's values.

Q. Dereferencing a pointer variable amounts to accessing the pointer variable's memory location, and using the memory address contained in that location to access memory a second time to read or write content at that address.

Q. Execution of this code,

```
int A[10];    int B[10];    A = B;    A[0] = 7;
```

causes array B's first integer memory location to contain the value 7.

Q. Execution of this code,

```
int A[10];    int *p;    p = A;   cout << p;   cout << A;
```

results in the same address being displayed twice because an array's name refers to a pointer variable.

Q. Execution of this code,

```
int A[10];    int *A;    p = A;   cout << &p;   cout << &A;
```

results in different addresses being displayed because an array's name refers to a pointer variable.

Q. Execution of this code,

```
int A[10];   cout << A;   cout << &A;
```

results in the same value displayed twice because an array's name does not refer to a pointer variable.

Q. Given this code,

```
void f() {   int A[10];   ...  A[0] = 7;   ...   }
```

as far as the compiler is concerned, the use of the symbol "A" in the statement "A[0] = 7" stands for, and is replaced by, the memory address of the first integer memory location of a contiguous set of 10 integer memory locations while the text "[0]" is replaced by "+0" in forming a memory address.

Q. In the previous question's example, the memory locations for the array's values are allocated on the stack at runtime; so, the memory address A refers to is an address in the area of memory used for stack operations.

Q. Function names are pointer variables with their own associated memory at runtime.

Q. When this example code is executed,

```
void f() {  ...  }       int main()  {    void (*p)();    p = f;    cout << p;  cout << f;  return 0;  }
```

p will display the same value as f because f is a pointer variable.

Q. If, in the previous code example, we replace the last occurrence of "p" with "&p" and "f" with "&f", different values will be displayed because p and f are different pointer variables.

Q. In we add this code to the previous example, " cout << f; cout << &f; ", we will see the same value twice because f is not a pointer variable.

Q. A function or method call results in the CPU's program counter (PC) being loaded with the memory address of the function's executable code. The next instruction will be fetched from that location.

Q. A class definition is not an area of memory, but at runtime a corresponding memory area is allocated to individual instances as needed.

Q. An instance of a class is an area of memory consisting of a collection of memory locations, one for each field of the class.

Q. An instance of a class is an area of memory consisting of a collection of memory locations, one for each method of the class.

Q. An instance of a class is an area of memory consisting of a collection of memory locations, one for each field and method of the class.

Q. There is exactly one copy of the executable body of each method of a class in memory at runtime. Instances share the same executable code, and the compiler generates code to jump to the same location whenever any class instance invokes the same method.

Q. Given the following code (assuming an appropriate class definition for class A),

```
int S() { /*-- scope S --*/ A a1, a2;  .... }
```

when execution enters scope S, two instances of class A (a1 and a2) are created, and one chunk of memory large enough to hold all field values defined in class A is allocated. References to a1 or a2 refer to that same chunk of memory.