Academic Challenge
Computer Science Test (State) - 2020

1. Convert the Hexadecimal number 0x67 to Octal.

   A. 147

   B. 103

   C. 137

   D. 01100111

   E. None of the above

2. Convert -16 to binary using Two's Complement notation.

   A. 11101111

   B. 00010000

   C. 11101111

   D. 11110000

   E. None of the above

3. On average, how many elements of a sorted array will need to be checked before a binary search finds the desired element? Assume the array is of length n.

   A. $n/2$

   B. $\log(n)$

   C. n

   D. $n-1$

   E. $n \cdot \log(n)$

4. Which of the following data structures would most-easily fascilitate locating the minimum number in a set of numbers, assuming the data structure is already set up and populated with data?

   A. Unsorted Array

   B. Min Heap

   C. Max Heap

   D. Binary Search Tree

   E. Queue

5. Which of the following is an example of a dynamically-typed language?

   A. C#

   B. C++

   C. Javascript

   D. Assembly

   E. Both A and B

Use the following code for questions 6 through 8:

```
#include <iostream>
using namespace std;

class Point
{
  private:
    int x;
    int y;
  public:
    Point(int x, int y) { this->x = x; this->y = y; };
    int getX() const    { return this->x; };
    int getY() const    { return this->y; };
};

class Vector : public Point
{
  public:
    Vector(int x, int y) : Point(x, y) { };
    int getMagnitude();
};

void Print(const Vector& v)
{
  cout << "(" << v.getX() << "," << v.getY() << ")";
}

int main()
{
  Vector a(2, 3);
  Point b(1, 5);
  Print(a);
  return 0;
}
```

6. Suppose we need to also create a `Vector3`, which would be a `Vector` in three dimensional space. Which of the following would allow us to reuse the Vector class for this new `Vector3` class?

   A. Polymorphism

   B. Compilation switches

   C. Inheritance

   D. Choice B or C

   E. None of the above

7. Which of the following would correctly overload the addition operator for `Vector`?

   A.
   ```
   Vector Vector::operator+(Vector& lhs, Vector& rhs)
   {
     Vector tmp(lhs.x + rhs.x, lhs.y + rhs.y);
     return tmp;
   }
   ```

   B.
   ```
   Vector Vector::operator+(Vector& rhs)
   {
     Vector tmp(this.x + rhs.x, this.y + rhs.y);
     return tmp;
   }
   ```

   C.
   ```
   Vector* Vector::operator+(Vector& rhs)
   {
     Vector tmp(this->x + rhs.x, this->y + rhs.y);
     return tmp;
   }
   ```

   D.
   ```
   void Vector::operator+(Vector& rhs)
   {
     this->x += rhs.x;
     this->y += rhs.y;
   }
   ```

   E.
   ```
   Vector Vector::operator+(Vector& rhs)
   {
     Vector tmp(this->x + rhs.x, this->y + rhs.y);
     return tmp;
   }
   ```

8. Which of the following would allow us to make the function call `Print(b);`?

   A. Polymorphism

   B. Function templates

   C. Inheritance

   D. The code would not compile

   E. None of the above

9. A Hamming Code is an example of which of the following?

   A. Parity function

   B. Hashing algorithm

   C. Error correcting and detection code

   D. Bluetooth locator beacon

   E. A specific type of Gray Code

10. Which of the following is not a source control system?

    A. git

    B. Team Foundation Version Control (TFVC)

    C. Subversion

    D. Merge Master

    E. SourceSafe

11. If two functions are overloaded versions of each other, which of the following is true?

    A. They have the same return type and parameter list

    B. They have the same function name and parameter list, but differ by their return type

    C. They are functions which call each other

    D. They exist in the same class

    E. They have the same function name

12. Suppose a computer were built such that instead of using Binary, it used a Base-3 numbering system as the fundamental method of storing and working with data. What would the left-shift operator do, if executed on such a computer?

    A. Divide the number by a multiple of 2

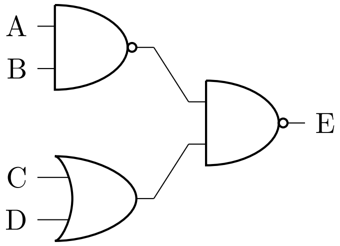
    B. Multiply the number by a multiple of 3

    C. Multiply the number by a multiple of 2

    D. Add a multiple of 3 to the number

    E. Divide the number by a multiple of 2

Use the following logic diagram for the next question:



13. Which of the following represents the minimum Sum of Products expression for the logic diagram?

    A. $AB + \neg(C \vee D)$

    B. $A \wedge B + \neg C \wedge \neg D$

    C. $\neg A \wedge B \vee \neg(C \vee D)$

    D. $A \wedge B \wedge \neg(C \vee D)$

    E. None of the above

Use the following instruction set definition and program for the next few questions:

Symbolic representation	**Description**
LOAD A(x)	Load the value at memory location x into register A
STOR A(x)	Copy the value from register A into memory location x
LOAD B(x)	Load the value at memory location x into register B
STOR B(x)	Copy the value from register B into memory location x
ADDM A(x)	Add the value at memory location x to register A
JMPN B(y)	Jump to program step y if the value of register B is less than zero
INCR B	Increment the value stored in register B by 1
DECR B	Decrement the value stored in register B by 1
STRL (x,d)	Store the literal value d to memory location x
NOOP	Don't do anything, and move to the next instruction

What is the following code doing?

```
0x01 STRL (0x10, 0x05)
0x02 NOOP
0x03 STRL (0x12, 0x00)
0x04 LOAD A(0x12)
0x05 LOAD B(0x10)
0x06 ADDM A(0x10)
0x07 DECR B
0x08 STOR B(0x10)
0x09 JMPN B(0x06)
0x0A STOR A(0x13)
```

14. What is in memory location `0x13` after the code executes?
    - A. 0
    - B. 5
    - C. 10
    - D. 15
    - E. NULL

15. What is the purpose of the data stored in memory location `0x12`?
    - A. Stores the initial value for the variable keeping track of the sum
    - B. Stores the index counter
    - C. Used to track when to exit the loop
    - D. Stores the result
    - E. The data is not used (NOOP)

16. How many times does the code on line `0x0A` execute?
    - A. 1
    - B. 5
    - C. 6
    - D. 10
    - E. 11

17. What is the purpose of the code on line `0x09`?
    - A. Decrements the value of the index counter
    - B. Stores the result to memory
    - C. Loops while A is not negative
    - D. Keeps track of the sum
    - E. Loops while B is not negative

18. If the code on line 0x02 were removed, how would the behavior of the code change?
    - A. The line numbers would be shifted, causing the jump statement to jump to the decrement line; therefore, resulting in a different output
    - B. The variables would be shifted in memory by 1, but the behavior would not change
    - C. The behavior would not change
    - D. The code would take longer to execute
    - E. None of the above

19. Is it possible to use a C++ reserved keyword as a variable name?
    - A. Yes, any reserved keyword can be used as a variable name
    - B. Yes, but only if using a recent C++ compiler
    - C. Yes, but only be removing the previous definition of that keyword
    - D. No, since there is no way to tell how the keyword is actually being used
    - E. No, since keywords are compiled as preprocessor directives

20. What is the Cyclomatic Time Complexity, expressed in Big-Oh notation, for Quick Sort in the worst case?
    - A. $\mathrm{O}(\log n)$
    - B. $\mathrm{O}(n^2)$
    - C. $\mathrm{O}(n)$
    - D. $\mathrm{O}(n \cdot \log n)$
    - E. $\mathrm{O}(n!)$

Use the following code for questions 21 through 24:

```
#include <iostream>
using namespace std;

const float PI = 3.14;

class Circle
{
  private:
    int radius;
  public:
    Circle(int radius)    { this->radius = radius; };
    int getRadius()       const { return radius; };
    int getCircumfrance() const { return 2 * PI * radius; };
    int getArea()         const { return PI * radius * radius; };
    bool operator<(Circle& rhs) { return radius < rhs.radius; };
};

class Cylinder : protected Circle
{
  private:
    int length;
  public:
    Cylinder(int radius, int length) : Circle(length)
                               { this->length = length; };
    int getRadius()      const { return Circle::getRadius(); };
    int getLength()      const { return this->length; };
    int getVolume()  const { return this->getArea() * this->length; };
    int getSurfaceArea() const { return this->getArea() * 2
                                + this->getCircumfrance() * length; };
};

template <class T>
T getMax(T ts[], int size) {
  T a = ts[0];
  for(int i = 1; i < size; i++)
    a = (a < ts[i]) ? ts[i] : a;
  return a;
}

int main()
{
  Cylinder c(2, 3);
  Cylinder c2[] = { Cylinder(1, 2), Cylinder(2, 3) };
  Circle   c3[] = { Circle(4), Circle(5) };

  return 0;
}
```

21. How many times does the default constructor for `Circle` get called?
    - A. 1
    - B. 2
    - C. 4
    - D. 5
    - E. None of the above

22. What principle of Object Oriented Programming best describes the `int Cylinder::getRadius()` function?
    - A. Inheritance
    - B. Abstraction
    - C. Encapsulation
    - D. Polymorphism
    - E. Overriding

23. Which of the following is a valid call to function `getMax`?
    - A. `getMax(c, 1);`
    - B. `getMax(c2, 1);`
    - C. `getMax(c3, 2);`
    - D. All of the above
    - E. Answers B and C

24. If the `const` keyword were removed from the function `int Circle::getArea()`, what would happen?
    - A. Nothing
    - B. A compile-time exception
    - C. A run-time exception
    - D. A compile-time warning
    - E. None of the above

25. Is it possible to overload the function call operator?
    - A. Yes, but only when overridden in classes
    - B. Yes, but only when overridden in structs
    - C. Yes
    - D. No
    - E. No, unless experimental compiler support is included

Use the following code for questions 26 through 28:

```
1  #include <iostream>
2
3  int main()
4  {
5    int a[2][3] = { { 3, 2, 7 },
6                    { 8, 1, 1 } };
7
8    int b[3][2] = { { 9, 1 },
9                    { 7, 5 },
10                   { 3, 0 } };
11
12   int c[2][2] = { { 0, 0 },
13                   { 0, 0 } };
14
15   for(int i = 0; i < 2; i++)
16     for(int j = 0; j < 2; j++)
17       for(int k = 0; k < 3; k++)
18         c[i][j] += a[i][k] * b[k][j];
19
20   for(int x = 0; x < 2; x++)
21   {
22     for(int y = 0; y < 2; y++)
23       std::cout << c[x][y] << " ";
24     std::cout << std::endl;
25   }
26
27   return 0;
28 }
```

26. What is printed to standard output?

    A. 62 82 \n13 13 \n

    B. 13 13 \n62 82 \n

    C. 0 0 \n0 0 \n

    D. 82 13 \n62 13

    E. 62 13 \n82 13 \n

27. How many times does the code on line 18 get executed?

    A. 6

    B. 9

    C. 12

    D. 15

    E. 18

28. What is the code calculating?

    A. The matrix-multiplication of two arrays

    B. The sum-of-products of boolean expressions

    C. The multiplicative inverse of two arrays

    D. The determinant of the matrices

    E. None of the above

Use the following code for questions 29 through 30:

```
int FibIterative(int a)
{
  int result = 0;
  int p0 = 0, p1 = 1;
  for(int i = 0; i < a; i++)
  {
    p0 = p1;
    p1 = result;
    result = p0 + p1;
  }

  return result;
}

int FibRecursive(int a)
{
  if(a == 0 || a == 1)
    return a;
  else
    return FibRecursive(a - 1) + FibRecursive(a - 2);
}
```

29. Which of the two functions would perform better for sufficiently large values for `a`?

    A. `FibIterative`, since it uses the Stack instead of the Heap

    B. `FibRecursive`, since it uses the Heap instead of the Stack

    C. `FibIterative`, since it calculates the result in O(n) time

    D. `FibRecursive`, since it calculates the result in O(n) time

    E. `FibRecursive`, since its implementation matches the definition of a Fibbonacci number more closely

30. How many times does `FibRecursive` get called, including the initial function call, with the call `FibRecursive(5)`?

    A. 9

    B. 8

    C. 7

    D. 6

    E. 5