2024 CCF 非专业级别软件能力认证第一轮

（CSP-J1）入门级 C+语言试题

认证时间：2024 年 9 月 21 日 09:30~11：30

考生注意事项：

- 试题纸有 12 页，答题纸共有 1 页，满分 100 分。 请在答题纸上作答 ，写在试题纸上的一律无效。
- 不得使用任何电子设备（如计算器、手机、电子词典等）或查任何书筹资料。

一、 单项选择题（共 15 题，每题 2 分，共计 30 分；每题有且仅有一个正确选项）

1．32 位 int 存储范围是（ ）

A.-2147483647~+2147483647

B.-2147483647~+2147483648

C.-2147483648~+2147483647

D.-2147483648~+2147483648

2．计算$(14_8-1010_2)*D_{16}-1101_2$ 的结果，并选择答案的十进制值：（ ）

A．13

B．14

C．15

D．16

3．某公司有 10 名员工，分为 3 个部门:A 部门有 4 名员工，B 部门有 3 名员工、C 部门有 3 名员工。现需要从这 10 名员工中选出 4 名组成一个工作组，且每个部门至少要有 1 人。问有多少种选择方式?（ ）

A．120

B．126

C．132

D．238

4．以下哪个序列对应数组 0 至 7 的 4 位二进制格雷码（Gray code）？（ ）

A．0000，0001，0011，0010，0110，0111，0101，1000

B．0000，0001，0011，0010，0110，0111，0100，0101

C．0000，0001，0011，0010，0100，0101，0111，0110

D．0000，0001，0011，0010，0110，0111，0101，0100

5．记 1Kb 位 1024 字节（byte），1MB 位 1024KB，那么 1MB 是多少二进制位（bit）？（ ）

A.1000000

B.1048576

C.8000000

D.8388608

6．以下哪个不是 C++中的基本数据类型？（ ）

A．int

B．float

C．struct

D．char

7．以下哪个不是 C++中的循环语句？（ ）

A．for

B．while

C．do-while

D．repeat-untill

8．在 C/C++中，(char)('a'+13)与下面的哪一个值相等（ ）

A．'m'

B．'n'

C．'z'

D. ‘3’

9. 假设有序表中有 1000 个元素，则用二分法查找元素 x 最多需要比较（ ）次

A.25

B.10

C.7

D.1

10. 下面哪一个不是操作系统名字（ ）

A. Notepad

B. Linux

C. Windows

D. macOS

11. 在无向图中，所有顶点的度数之和等于（ ）

A. 图的边数

B. 图的边数的两倍

C. 图的定点数

D. 图的定点数的两倍

12. 已知二叉树的前序遍历为[A,B,D,E,C,F,G],中序遍历为[D,B,E,A,F,C,G],求二叉树的后序遍历的结果是（ ）

A. [D,E,B,F,G,C,A]

B. [D,E,B,F,G,A,C]

C. [D,B,E,F,G,C,A]

D. [D,B,E,F,G,A,C]

13. 给定一个空栈，支持入栈和出栈操作。若入栈操作的元素依次是 1 2 3 4 5 6,其中 1 最先入栈，6 最后入栈，下面哪种出栈顺序是不可能的（ ）

A.6 5 4 3 2 1

B.1 6 5 4 3 2

C.2 4 6 5 3 1

D.1 3 5 2 4 6

14. 有 5 个男生和 3 个女生站成一排，规定 3 个女生必须相邻，问有多少种不同的排列方式？（ ）

A. 4320 种

B. 5040 种

C. 3600 种

D. 2880 种

15. 编译器的主要作用是什么（ ）？

A. 直接执行源代码

B. 将源代码转换为机器代码

C. 进行代码调试

D. 管理程序运行时的内存

二、 阅读程序（程序输入不超过数组或字符串定义的范围；判断题正确填√，错误填 ×；除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分。

（1）

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

bool isPrime(int n) {
    if (n <= 1) {
        return false;
    }
    for (int i = 2; i * i <= n; i++) {
            if (n % i == 0) {
                    return false;
            }
    }
    return true;
}

int countPrimes(int n) {
    int count = 0;
    for (int i = 2; i <= n; i++) {
            if (isPrime(i)) {
                count++;
            }
    }
    return count;
}

int sumPrimes(int n) {
    int sum = 0;
    for (int i = 2; i <= n; i++) {
            if (isPrime(i)) {
                    sum += i;
            }
    }
    return sum;
}

int main() {
    int x;
    cin >> x;
```

```
    cout << countPrimes(x) << " " << sumPrimes(x) << endl;
    return 0;
}
```

- 判断题

16.当输入为“10”时，程序的第一个输出为“4”，第二个输出为“17”。（ ）

17.若将 isPrime(i)函数种的条件i * i <= n 改为 i<=n/2,输入“20”时，countPrimes(20)的输出将变为“6”（ ）

18. sumPrimes 函数计算的是从 2 到 n 之间的所有素数之和（ ）

- 单选题

19.当输入为“50”时，sumPrimes(50)的输出为（ ）

A.1060

B.328

C.381

D.275

20.如果将 for(int i=2;i*i<=n;i++)改为 for(int i=2;i<=n;i++),输入“10”时，程序的输出（ ）

A.将不能正确计算 10 以内素数个数及其和

B.仍然输出“4”和“17”

C.输出“3”和 10

D.输出结果不变，但运行时间更短

（2）

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

int compute(vector<int> &cost) {
    int n = cost.size();
    vector<int> dp(n + 1, 0);
    dp[1] = cost[0];
```

```
09      for (int i = 2; i <= n; i++) {
10              dp[i] = min(dp[i - 1], dp[i - 2]) + cost[i - 1];
11      }
12      return min(dp[n], dp[n - 1]);
13  }
14
15  int main() {
16      int n;
17      cin >> n;
18      vector<int> cost(n);
19      for (int i = 0; i < n; i++) {
20          cin >> cost[i];
21      }
22      cout << compute(cost) << endl;
23      return 0;
24  }
```

- 判断题

21. 当输入的 cost 数组为{10，15，20}时，程序的输出为 15（　）

22. 如果将 dp[i-1]改为 dp[i-3]，程序可能会产生编译错误（　）

23.（2 分）程序总是输出 cost 数组中的最小的元素（　）

- 单选题

24. 当输入的 cost 数组为 {1,100,1,1,1,100,1,1,100,1} 时，程序的输出为（　）

A.6

B.7

C.8

D.9

25. (4 分) 如果输入的 cost 数组为{10，15，30,5,5,10，20}，程序的输出为（　）。

A. "25"

B. "30"

C. "35"

D. "40"

26. 若将代码中的 min(dp[i-1],dp[i-2])+cost[i-1]修改为 dp[i-1]+cost[i-2]，输入 cost 数组为 {5,10,15}时，程序的输出为（　）

A. 10

B. 15

C. 20

D. 25

（3）

```
01 #include <iostream>
02 #include <cmath>
03 using namespace std;
04
05 int customFunction(int a, int b) {
06     if (b == 0) {
07             return a;
08     }
09     return a + customFunction(a , b - 1);
10 }
11
12 int main() {
13     int x, y;
14     cin >> x >> y;
15     int result = customFunction(x, y);
16     cout << pow(result, 2) << endl;
17     return 0;
18 }
```

- 判断题

27. 当输入为“2 3”时，customFunction（2,3）的返回值为“64”。（　）

28. 当 b 为负数时，customFunction（a，b）会陷入无限递归。（　）

29. 当 b 的值越大，程序的运行时间越长。（　）

- 单选题

30.当输入为“5 4”时，customFunction（5,4）的返回值为（　）。

A.5

B.25

C.250

D.625

31. 如果输入 x = 3 和 y = 3，则程序的最终输出为（ ）

A."27”

B."81”

C."144”

D."256

32. （4 分）若将 customFunction 函数改为“return a + customFunction（a-1，b-1）；并输入“3 3”，则程序的最终输出为（ ）。

A.9

B.16

C.25

D.36

三、 完善程序（单选题，每小题 3 分，共计 30 分）

（1）(判断平方数) 问题：给定一个正整数 n，判断这个数是否为完全平方数，即存在一个正整数 x 使得 x 的平方等于 n

试补全程序。

```
01 #include <iostream>
02 #include <vector>
03 using namespace std;
04 bool isSquare(int num){
05     int i = ① ;
06     int bound = ② ;
07     for(;i<=bound;++i){
08         if( ③ ){
09             return ④ ;
10         }
11     }
12     return ⑤ ;
13 }
```

```
14 int main(){
15     int n;
16     cin>>n;
17     if(isSquare(n)){
18         cout<<n<<" is a Square number"<<endl;
19     }else{
20         cout<<n<<" is not a Square number"<<endl;
21     }
22     return 0;
23 }
```

33. ①处应填（　）

A. 1　　B. 2　　C. 3　　D. 4

34. ②处应填（　）

A. (int) floor(sqrt(num)-1)　　B. (int)floor(sqrt(num))

C. floor(sqrt(num/2))-1　　D. floor(sqrt(num/2))

35. ③处应填（　）

A. num=2*i　　B. num== 2*i

C. num=i*i　　D. num==i*i

36. ④处应填（　）

A. num= 2*i　　B. num==2*i　　C. true　　D. false

37. ⑤处应填（　）

A. !(num=2*i)　　B. num!=2*i　　C. true　　D. false

（2）（汉诺塔问题）给定三根柱子，分别标记为 A、B 和 C。初始状态下，柱子 A 上有若干个圆盘，这些圆盘从上到下按从小到大的顺序排列。任务是将这些圆盘全部移到柱子 C 上，且必须保持原有顺序不变。在移动过程中，需要遵守以下规则：

1. 只能从一根柱子的顶部取出圆盘，并将其放入另一根柱子的顶部。
2. 每次只能移动一个圆盘

3. 小圆盘必须始终在大圆盘之上。

试补全程序。

```
01 #include <bits/stdc++.h>
02 using namespace std;
03 void move(char src, char tgt) {
04     cout << "从柱子" << src << "挪到柱子上" << tgt << endl;
05 }
06 void dfs(int i, char src, char tmp, char tgt) {
07     if(i == ① ) {
08         move( ② );
09         return;
10     }
11     dfs(i-1, ③ );
12     move(src, tgt);
13     dfs( ⑤ , ④ );
14 }
15 int main() {
16     int n;
17     cin >> n;
18     dfs(n, 'A', 'B', 'C');
19     return 0;
20 }
```

38. ①处应填（　）

A.0　　B.1　　C.2　　D.3

39. ②处应填（）

A. src,tmp　　B. src,tgt　　C. tmp,tgt　　D. tgt,tmp

40. ③处应填（）

A. src,tmp,tgt　　B. src, tgt, tmp

C. tgt, tmp, src　　D. tgt, src, tmp

41. ④处应填（）

A. src, tmp, tgt　　B. tmp,src, tgt

C. src, tgt,tmp　　D. tgt,src,tmp

42. ⑤处应填（）

A. 0　　B. 1　　C. i-1　　D. i