Suppose you declare int count = 5; which of the following is true?

- A. &count is the address of count
- B. &count is 5
- C. *count is the address of count
- D. *count is 5

2. Suppose you declare int count = 5 and int *pCount = &count; which of the following is true?

- A. *count is the address of count
- B. &count is 5
- C. *pCount is 5
- D. pCount contains the address of count

3. If you declare a variable double d = 5.5 and compiler stores it in the memory starting with address 04BFA810, then &d is _____.

- A. 5
- B. 5.5
- C. 0
- D. unknown
- E. 04BFA810

4. Why the following pointer variable declaration is wrong?

```
int area = 1;
double *pArea = &area;
```

- A. double *pArea = &area should be double *pArea = area;

B. the type of variable does not match the type of the pointer.

C. double *pArea = &area should be float *pArea = area;

D. double *pArea = &area should be int *pArea = area;

5. Which of the following statements are true?

A. A local variable is assigned an arbitrary value if you don?t initialize it.

B. A local pointer is assigned an arbitrary value if you don?t initialize it.

C. An array element is assigned an arbitrary value if you don?t initialize it.

D. Dereferencing a pointer that is not initialized could cause fatal runtime error or it could accidentally modify important data.

6. Suppose int list[6] = {11, 12, 13, 14, 15, 16}; Is *list the same as list[0]?

A. yes

B. no

7. Suppose you declare an array double list[] = {1, 3.4, 5.5, 3.5} and compiler stores it in the memory starting with address 04BFA810. Assume a double value takes eight bytes on a computer. &list[1] is _____.

A. 04BFA810

B. 04BFA818

C. 1

D. 3.4

8. Suppose you declare an array double list[] = {1, 3.4, 5.5, 3.5}. &list[1] is same as _____.

A. list

- B. list + 1
- C. list + 2
- D. list[0]
- E. list[1]

9. Suppose you declare an array double list[] = {1, 3.4, 5.5, 3.5}. *(list + 1) is same as _____.

- A. *list
- B. *list + 1
- C. *list + 2
- D. list[0]
- E. list[1]

10. What is the output of the following code?

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

int main()
{
  int list[] = {10, 20, 30, 40};
  cout << *(list + 1) << " " << *list + 1 << endl;

  return 0;
}
```

- A. 10 10
- B. 20 20
- C. 30 30
- D. 20 11

11. What is the output of the following code?

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

int main()
{
  int list[] = {1, 1, 1, 1};
  *(list) = *(list) + 1;
  *(list + 1) = *(list + 1) + 2;
  *(list + 2) = *(list + 2) + 3;
  *(list + 3) = *(list + 3) + 4;
  cout << list[0] << " " << list[3] << endl;

  return 0;
}
```

- A. 1 2
- B. 2 2
- C. 3 4
- D. 3 5
- E. 2 5

12. Suppose you defined

```
int list1[4], list2[4];
int *p1, *p2;
```

Which of the following statements are correct?

- A. p1 = list1;
- B. p1 = p2;
- C. list1 = p1;
- D. list1 = list2;

13. Analyze the following code.

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

```
int main()
{
  char *p;
  cout << "Enter a string: ";
  cin >> p;

  cout << p << endl;

  return 0;
}
```

A. If you run the program and enter abc, nothing will be displayed. The program runs without errors.

B. If you run the program and enter abc, abc will be displayed.

C. If you run the program and enter abc, unpredictable characters will be displayed.

D. If you run the program and enter abc, a runtime error will occur, because p is used without being initialized.

14. Analyze the following code.

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

int main()
{
  char t[10];
  char * p = t;
  cout << "Enter a string: ";
  cin >> p;

  cout << p << endl;

  return 0;
}
```

A. If you run the program and enter abc, nothing will be displayed. The program runs without errors.

B. If you run the program and enter abc, abc will be displayed.

C. If you run the program and enter abc, unpredictable characters will be

displayed.

D. If you run the program and enter abc, a runtime error will occur, because p is being used without initialized.

15. What is the output of the following code?

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

void swap(int *pValue1, int *pValue2)
{
  cout << "swap 1 invoked" << endl;
}

void swap(int &pValue1, int &pValue2)
{
  cout << "swap 2 invoked" << endl;
}

int main()
{
  int num1 = 1;
  int num2 = 2;

  swap(&num1, &num2);

  return 0;
}
```

A. swap 1 invoked

B. swap 2 invoked

C. The program has a runtime error because swap is declared multiple times.

D. The program has a compile error because swap is declared multiple times.

16. What is the output of the following code?

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

```
void swap(int *pValue1, int *pValue2)
{
  cout << "swap 1 invoked" << endl;
}

void swap(int &pValue1, int &pValue2)
{
  cout << "swap 2 invoked" << endl;
}

int main()
{
  int num1 = 1;
  int num2 = 2;

  swap(num1, num2);

  return 0;
}
```

- A. swap 1 invoked
- B. swap 2 invoked
- C. The program has a runtime error because swap is declared multiple times.
- D. The program has a compile error because swap is declared multiple times.

17. What is the output of the following code?

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

void swap(int pValue1, int pValue2)
{
  cout << "swap 1 invoked" << endl;
}

void swap(int &pValue1, int &pValue2)
{
  cout << "swap 2 invoked" << endl;
}
```

```
int main()
{
  int num1 = 1;
  int num2 = 2;

  swap(num1, num2);

  return 0;
}
```

A. swap 1 invoked

B. swap 2 invoked

C. The program has a runtime error because swap is declared multiple times.

D. The program has a compile error because swap(num1, num2) could match either swap(int pValue1, int pValue2) or swap(int &pValue1, int &pValue2).

18. Which of the following declaration is correct?

A. int *pValue = new double;

B. int *pValue = new int;

C. double *pValue = new double;

D. double *pValue = new int;

19. Suppose list is declared as follows:

int *list = new int[10];

How should you destroy list?

A. delete list;

B. delete *list;

C. delete [] list;

D. delete [] *list;

20. Does the following code cause a memory leak?

```
int *pValue = new int;
*pValue = 45;
pValue = new int;
delete pValue;
```

A. yes

B. no

答案：
1. A
2. CD
3. E
4. BD
5. ABD
6. A
7. B
8. B
9. E
10. D
11. E
12. AB
13. D
14. B
15. A
16. B
17. D
18. BC
19. C
20. A