Question 1 of 20 4.5 Points

The program:

```
#include <stdio.h>
void main( )
{
for( int i=1; i<4; i++)
{
static int j=1; int k=1;
printf("\n i=%d j=%d k=%d", i,j,k);
j++; k++;
}
}
```

displays:

A. 1 1 1
2 2 1
3 3 1

B. 1 1 1
2 1 1
3 1 1

C. 1 1 1
2 2 2
3 3 3

D. 1 1 1
2 1 2
3 1 3

E. other combination of values

Reset Selection

Question 2 of 20 4.5 Points

Which of the following elements isn't part of the „strong name" concept:

- A. version number
- B. culture with which it is associated
- C. public key
- D. digital signature
- E. security policy

Reset Selection

Question 3 of 20 4.5 Points

Specify what displays the program below:

```
#include <stdio.h>
void main()
{
int *aux = new int,i, **tabela = new int* [3];
for(i=0; i<3; i++)
{ *aux=i; tabela[i] = aux; }
for(i=0; i<3; i++)
printf("%d ", *tabela[i]);
}
```

- A. 1 2 3
- B. 0 1 2
- C. 2 2 2
- D. 1 1 1
- E. 3 3 3

Reset Selection

Question 4 of 20 4.5 Points

```
#include <stdio.h>
void main ( )
{
int x = 7,y = 8, &ri = x; printf("\n %d ",ri);
ri = y; y = 10; printf("\n %d ",ri);
}
```

The sequence above:

- ○ A. is wrong, assignment ri = y; is not permitted;
- ○ B. displays 7 10 , because ri is referring now y;
- ● C. displays 7 8 , because a reference can't be changed, only be dereferred as variable;
- ○ D. is wrong, a reference can't be displayed;
- ○ E. displays 10 10, the last reference initialization is that retained by the compiler.

Reset Selection

Question 5 of 20 4.5 Points

An attack that only allows the attacker to inhibit a service, without gaining anything else, is known as:

- A. Denial of Service (DoS)
- B. Service Linking
- C. Watermarking
- D. Obfuscation
- E. Machine „owning”

Reset Selection

Question 6 of 20 4.5 Points

What is the best way to secure and reduce as size the programs- based on dynamic link libraries programs (DLLs)?

- A. RSA Encryption
- B. Assembly Linking
- C. Watermarking
- D. Library Bypass Obfuscation
- E. Fingerprinting

Reset Selection

Question 7 of 20 4.5 Points

The program becomes easier decompiled, because Visual C#.NET is compiled firstly into an intermediate language ?

- A. No, because the intermediate language is more difficult to understand by an attacker
- B. Yes, because restoring the programming structures is much easier
- C. No, because the intermediate language is platform dependent running
- D. Yes, because the intermediate language allows running on a virtual machine
- E. C#.NET language is scripting type and is not compiled, but is interpreted

Reset Selection

Question 8 of 20 4.5 Points

Obfuscators alter the source code and make safer programs, but cannot be debugged.

- (●) A. Yes, because it cannot restore the original context of the program
- ( ) B. No, because obfuscators change the source code, but keep some original information too;
- ( ) C. Yes, because searching in tables of association is no longer possible
- ( ) D. Obfuscated programs cannot be repaired
- ( ) E. Obfuscation doesn't make safer programs

Reset Selection

Question 9 of 20 4.5 Points

```
#include <iostream.h>
class dst { public: int x; };
class srs
{
private: int x;
public:
srs(int a):x(a){}
operator dst() { dst d; d.x = x; return d; }
};

void main()
{
srs s(1); dst d=(dst)s; cout << d.x;
}
```

What statement concerning the security of the program is true:

- A. Bad casting gives access to private data
- B. Object oriented programming is always secure
- C. Casting from srs to dst is not permitted
- D. Operator overloading is always unsecure
- E. There is no operator<< overloading, but it is called

Reset Selection

Question 10 of 20 4.5 Points

Reducing the risk of an attack on a program "managed code" is done by:

- ○ A. Insertion of additional validations
- ○ B. Use of exceptions mechanism
- ○ C. The use of privileged accounts
- ● D. Running under a virtual machine or using code obfuscation
- ○ E. Impersonation

Reset Selection

Question 11 of 20 4.5 Points

```
#include <stdio.h>
void main()
{
char *pt="01234567890"; puts(pt+2); getchar();
}
```

displays:

- A. 1234567892
- B. 01234567892
- C. 23456789
- D. 234567890
- E. 123456789

Reset Selection

```
namespace RaceConditions_Thread
{
   class ProcessData
   {
     int result = 0;
   public void Process()
   {
      Thread worker1 = new Thread(Work1),
           worker2 = new Thread(Work2);
      worker1.Start();    worker2.Start();
      Console.WriteLine(result);
   }
   void Work1() { result = 1;   }
   void Work2() { result = 2;   }

    public static void Main(string[] args)
     {
        ProcessData p = new ProcessData();
        p.Process(); Console.Read();
     }

   }
}
```

displays:

- ○ A. 0, initialization value for result
- ○ B. 1, first thread assigned the result
- ○ C. 2, last thread assigned the result
- ● D. 0,1,2 depending of running context(threads scheduling)
- ○ E. a residual value

Reset Selection

Question 13 of 20 4.5 Points

```
#include <stdio.h>
#include <string.h>
void main()
{
int count;
printf( "1234567890123456%n7890", &count );
printf( "\n %d ", count );
getchar();
}
```

printf("\n %d ", count); displays:

- ○ A. 20
- ○ B. the address of count variable
- ● C. 16
- ○ D. 1
- ○ E. other value

Reset Selection

Question 14 of 20 4.5 Points

```
#include <stdio.h>
void main()
{
char buf[]= "%p %p %p %p %p %p %p %p";
printf(buf);
getchar();
}
```

The sequence above:

- ○ A. is wrong, because the printf call doesn't get the formatting information;
- ○ B. is wrong because the printf call doesn't get the variables for displaying;
- ● C. displays variables or addresses located on the stack;
- ○ D. displays the variables or addresses from heap;
- ○ E. displays the variables or addresses from registries

Reset Selection

Question 15 of 20 4.5 Points

The technique for hiding information in a program which allow identify the way that an unauthorized user has come into the program possession, is called:

- A. RSA Encryption
- B. Assembly Linking
- C. Watermarking
- D. Library Bypass Obfuscation
- E. Code managing

Reset Selection

Question 16 of 20 4.5 Points

What displays the program below, guided you by the comments in the source text

```
#include <stdio.h>
void function(int a, int b, int c)
{
char buffer1[5]="ABCD"; char buffer2[10]="123456789";
char *ret;
ret = buffer1 + 16;
// ret points on adr where the return address is kept
(*ret) += 8;
}
void main()
{
int x; x = 0;
function(1,2,3);
x = 1;
printf("%d\n",x); getchar();
}
```

- A. 1
- B. 16
- C. 0
- D. 8
- E. 4

Reset Selection

Question 17 of 20 4.5 Points

Which of the following options IS NOT a remedy to SQLInjection vulnerability ?

- A. writing between single quotation marks the variable part, given by user;
- B. using of stored procedures;
- C. avoiding the query of remote servers;
- D. using the parameters for transmission the values to server;
- E. validation of the values entered by user

Reset Selection

Question 18 of 20 4.5 Points

Stack inspection is a technique by which:

- ○ A. it is prevented the call of a privileged function by a non privileged function;
- ● B. it is eliminated the possibility that a non privileged function to get special access rights calling a privileged function;
- ○ C. caller function is protected from the vulnerabilities of unsafe code from the called function;
- ○ D. variables or addresses on the stack are displayed;
- ○ E. the data on the stack are coded for not being consulted by a malicious code.

Reset Selection

Question 19 of 20 4.5 Points

What is the name of the arbitrary code call made by a hacker through exploiting the context of security from an assembly level ?

- ○ A. Direct Manipulation
- ○ B. Virtual Attack
- ○ C. Confused Deputies
- ● D. Code Injection
- ○ E. Spoofing

Reset Selection

Question 20 of 20 4.5 Points

Which of the following statements are not related to "strong type" concept

- A. clear delimitation of operations that are allowed on a type of data
- B. defining all accepted conversions for a data type
- C. the knowledge of types since compilation stage
- D. initializing all variables before use
- E. prohibiting any conversion from a data type to another.

Reset Selection