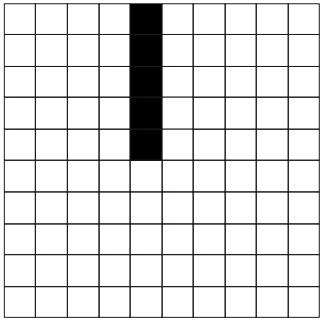
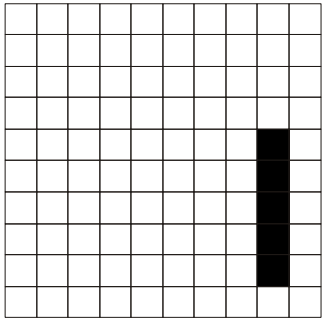
1. Co będzie wynikiem wykonania poniższych instrukcji?

```
g2d.getTransform().scale(1, -1);
g2d.getTransform().translate(4, -8);
g2d.drawLine(4, 0, 4, 4);
```

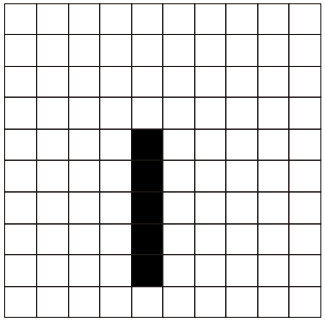
a)



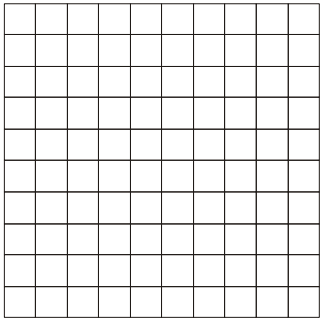
b)



c)



d)



2. Jaki będzie wynik kompilacji i wykonania programu zawierającego linię:

```
B b = new B();
```

dla klas zdefiniowanych jak niżej?

```
abstract class A{
  A(){ m(); }
  void m() { System.out.println("A"); }
}
class B extends A{
    B(){ super(); }
    void m(){ System.out.println("B"); }
}
```

a) błąd kompilacji (nie można wywoływać konstruktora klasy abstrakcyjnej)
b) błąd kompilacji (nie można przysłaniać metody klasy abstrakcyjnej)
c) `A`
d) `B`

3. Jaki będzie wynik kompilacji i wykonania linijki programu:

```
A a = new B();
```

dla klas zdefiniowanych jak niżej?

```
abstract class A{
  A(){ m(); }
  abstract void m();
}
class B extends A{
    B(){ super(); }
    void m(){ System.out.println("B"); }
}
```

a) błąd kompilacji (nie można wywoływać konstruktora klasy abstrakcyjnej)
b) błąd kompilacji (nie można przysłaniać metody klasy abstrakcyjnej)
c) błąd wykonania (wyjątek `NullPointerException`)
d) `B`

4. Jeśli metoda `Hello()` zwraca wartość typu `void` i nie wymaga podania żadnych argumentów, to jej deklaracja wygląda następująco:

a) `Hello();`
b) `Hello(void);`
c) `void Hello();`
d) `void Hello(void);`

5. W wyrażeniu:

```
public XXXX extends something1, something2
```

a) `XXX` powinno być interfejsem, `something1` oraz `something2` zaś interfejsami być nie muszą
b) `XXX` powinno być klasą, `something1` oraz `something2` muszą być interfejsami
c) `XXX`, `something1` oraz `something2` muszą być interfejsami
d) wyrażenie zawsze będzie niepoprawne, ponieważ Java nie dopuszcza dziedziczenia wielokrotnego

6. Jaki będzie wynik kompilacji i działania poniższego fragmentu programu?

```
Integer ten=new Integer(10);
Long nine=new Long (9);
System.out.print(ten + nine);
int i=1;
System.out.println(i + ten);
```

a) błąd kompilacji
b) `109110`
c) `1911`
d) `10911`

7. Co można powiedzieć dla klasy o poniższej definicji?

```
public class A {
    A() {}
    }
```

a) do klasy `A` można się odwoływać spoza pakietu, w którym jest zdefiniowana
b) nie można stworzyć obiektu klasy `A` poza pakietem, w który jest ona zdefiniowana
c) nie można rozszerzać klasy `A` poza pakietem, w który jest ona zdefiniowana
d) do klasy `A` można się odwoływać, tworzyć jej obiekty oraz ją rozszerzać gdziekolwiek

8. Jaki będzie wynik kompilacji i wykonania poniższego programu?

```
public class A {
      private void m1() throws Exception {
          throw new RuntimeException();
      }
      public void m2() {
        try { m1(); }
         catch(RuntimeException e) {
            System.out.print("RE "); }
        catch(Exception e) {
            System.out.print("E "); }
        }
      public static void main(String args[]) {
              A a = new A(); a.m2(); }}
```

a) błąd kompilacji
b) `RE`
c) `E`
d) `RE E`

9. Jaki będzie wynik kompilacji i wykonania poniższego programu?

```
public class Watki implements Runnable {
    public void run() { while(true) {} }

    public static void main(String args[]) {
        Watki w1 = new Watki();
        Watki w2 = new Watki();
        Watki w3 = new Watki();
        w1.run(); w2.run(); w3.run(); } }
```

a) błąd kompilacji (nieosiągalne `w2.run()`)
b) powstaną 3 nie kończące się wątki nie będące demonami
c) powstanie jeden nie kończący się wątek nie będący demonem
d) program szybko zakończy (ze względu na zakończenie metody `main`)

10. Kiedy kończy się działanie programu Javy?

a) po zakończeniu się metody `main`
b) kiedy wszystkie wątki nie będące demonami stworzonymi w aplikacji zakończą swoje działanie
c) kiedy wszystkie wątki – demony stworzone w aplikacji zakończą swoje działanie
d) kiedy wątek wykona `System.exit();`

11. Jaki będzie wynik kompilacji i działania poniższego programu?

```
public class Watek extends Thread {
      static String s = "a";
      public static void main(String argv[]){
            Watek w = new Watek();
            w.m(s); System.out.println(s); }
      public void m(String s){
             s = s + " b";
             start(); }
      public void run(){
         for(int i=0;i  <  3; i++)
             s = s + " " + i; } }
```

a) błąd kompilacji
b) `a b`
c) `a b 0 1 2 3`
d) `a` lub `a 0` lub `a 0 1` lub `a 0 1 2`

12. Jaki będzie wynik kompilacji i działania poniższego programu?

```
public class MyClass {
    public static void main(String args[]) {
        System.out.println("A");
    }
    public static void  main(char args[]) {
        System.out.println('B');
    }
}
```

a) błąd kompilacji (podwójna deklaracja metody `main`)
b) błąd wykonania (wyjątek `RuntimeException`)
c) `A`
d) `B`

13. Co jest poprawną deklaracją stałej w klasie?

a) `const int LINEFEED=10;`
b) `static final int LINEFEED=10;`
c) `int LINEFEED=10;`
d) `final int LINEFEED=10;`

14. Jaki będzie wynik kompilacji i działania poniższego programu?

```
abstract class M1 {
    static int i;
    abstract void amethod();
}
public class M2 extends M1 {
    public static void main(String argv[]) {
        int[] ar=new int[5];
        for(i=0;i < ar.length;i++)
            System.out.println(ar[i]);
    }
}
```

a) błąd kompilacji (referencja do `ar` użyta przed jej inicjalizacja)
b) błąd kompilacji (klasa `M2` powinna być zadeklarowana jako `abstract`)
c) błąd wykonania (zgłoszony wyjątek `IndexOutOfBoundes`)
d) pięć linii z wypisanym `0`

15. Jaki będzie wynik działania poniższego kodu?

```
int i=1;
switch (i) {
    case 0: System.out.print("zero "); break;
    case 1: System.out.print("one ");
    case 2: System.out.print("two ");
    default: System.out.print("default ");
}
```

a) `one`
b) `one default`
c) `one two default`
d) `default`

16. Jaki będzie wynik kompilacji i działania poniższego programu?

```
public class MyClass {
     static int i;
     public static void main(String argv[]){
    System.out.println(i); }
}
```

a) błąd kompilacji (zmienna i może nie zostać zainicjalizowana)
b) `1`
c) `0`
b) `null`

17. Jaki będzie wynik kompilacji poniższego kodu i wywołania wirtualnej maszyny z argumentem `C` ?

```
class B {}
class Sub extends B {}
class C {
  public static void main(String argv[]) {
       B b=new B ();
       Sub s=(Sub) b;}
   }
```

a) błąd kompilacji
b) błąd wykonania (wyjątek: `ClassCastException`)
c) błąd wykonania (wyjątek: `NoSuchMethodError`)
d) poprawna kompilacja i wykonanie programu

18. Jaki będzie wynik kompilacji i działania poniższego programu?

```
import javax.swing.*;
import java.awt.*;
public class B extends JFrame {
  B(){
    JButton HB=new JButton("Hello");
    JButton BB=new JButton("Bye");
    getContentPane().setLayout(new GridLayout(2,2));
    getContentPane().add(HB);
    getContentPane().add(BB);
    setSize(300,300); setVisible(true);  }
    public static void main(String argv[]){
    B b =new B(); }
   }
```

a) błąd kompilacji (funkcja `add` przy `GridLayout` powinna mieć dwa argumenty)

b) jeden przycisk `Bye` zajmujący całą ramkę

c) dwa przyciski w poziomie, `Hello` po lewej stronie, `Bye` po prawej

d) dwa przyciski w pionie, `Hello` na górze, `Bye` na dole

19. Jaki będzie wynik kompilacji i działania poniższego fragmentu programu?

```
public class A{
  static int j=20;
  public static void main(String argv[]) {
       int i=10;
       A a = new A();
       a.m(i);
       System.out.print(i);
       System.out.print(" ");
       System.out.println(j);    }
  public void m(int x){
       x=x*2;   j=j*2; }
  }
```

a) błąd kompilacji (zły parametr metody `m`)
b) `20 40`
c) `10 40`
d) `10 20`

20. Jaki będzie wynik kompilacji i działania poniższego programu?

```
public class A{
    public static void main(String argv[]) {
        int i, j;
 etykieta1: for (i=1;i <3;i++)
    etykieta2: for(j=1; j<3; j++) {
            if (j==2) continue etykieta1;
              System.out.print("i=" + i + " j=" +j +" ");  } } }
```

a) błąd kompilacji
b) `i=1 j=1 i=2 j=1`
c) `i=1 j=1 i=2 j=1 i=2 j=2`
d) `i=1 j=1 i=2 j=1 i=2 j=2  i=3 j=1`

21. Jaki będzie wynik kompilacji i działania poniższego programu?

```
public class A {
    final int i ;
    public static void main(String[] arguments) {
            System.out.println(new A().i); }
}
```

a) błąd kompilacji (nie zainicjalizowana zmienna `i`)
b) błąd kompilacji (zmiennej finalnej `i` nie wolno używać w metodach statycznych)
c) błąd wykonania
d) `0`

22**. Które z poniższych twierdzeń są prawdziwe?**

a) klasy zewnętrzne mogą być deklarowane jako `static`
b) klasy wewnętrzne mogą być deklarowane jako `static`
c) klasy lokalne mogą być deklarowane jako `static`
d) klasy anonimowe mogą być deklarowane jako `static`

23. Jaki będzie wynik kompilacji i wykonania następującego programu?

```
public class A {
   public static void main(String argv[]) {
         A r = new A();
         r.met(r); }
   public void met(A r) {
         int i=1;
         multi(r);
         System.out.println(i);    }
   public void multi(A r) {
         r.i = r.i*2;   }
}
```

a) błąd kompilacji
b) błąd wykonania
c) 1
d) 2

24. Które z klas pakietu `java.net` używane są przy tworzeniu gniazd?

a) `DatagramSocket` i `StreamSocket`
b) `MulticastSocket` i `UDPSocket`
c) `ServerSocket` i `Socket`
d) `TCPSocket` i `BroadcastSocket`

25. Które z poniższych poleceń nie utworzy połączenia z maszyną, na której uruchomiono program?

a) `new Socket("localhost", 80);`
b) `new Socket(null, 80);`
c) `new Socket("127.0.0.1", 80);`
d) `new Socket(new byte[]{127,0,0,1}, 80);`

26. Co z poniższego jest prawdziwe dla gniazd implementowanych w Javie?

a) dane są czytane z gniazda za pomocą metody `read()` instancji klasy gniazda
b) dane są czytane z gniazda za pośrednictwem metod odczytu ze strumienia wejściowego, który jest tworzony odpowiednią metodą gniazda
c) dane są zapisywane do gniazda za pośrednictwem metod zapisu strumienia wyjściowego, który jest tworzony odpowiednią metodą gniazda
d) dane są zapisywane do gniazda za pomocą metody `write()` instancji klasy gniazda

27. Które z deklaracji funkcji makeMI są poprawne dla poniższego interfejsu?

```
interface MI { int getI(); }
```

a)
```
MI makeMI(int i) {
    return new MI() { public int getI() { return i; }}; }
```

b)
```
MI makeMI(final int i) {
    return new MI() { public int getI() { return i; }}; }
```

c)
```
MI makeMI(int i) {
    class C implements MI { public int getI() { return i;}}
    return new C(); }
```

d)
```
MI makeMI(int i) { return new C(i); }
```
przy czym zewnętrzna klasa C zdefiniowana jest następująco:
```
class C implements MI {
    int j;
    C(int i) { j = i; }
    public int getI() { return j; }
  }
```

28. Jaki będzie wynik kompilacji i wykonanie poniższego kodu dla standardowo skonfigurowanej wirtualnej maszyny Java

```
public class A {
   public static void main(String argv[]) {
    double[] b[] = new double[20000][20000];
    b[0][0] = 1.0; }
}
```

a) błąd kompilacji
b) poprawna kompilacja, wykonanie zakończone wyrzuceniem wyjątku `NullPointerException`
c) poprawna kompilacja, wykonanie zakończone wyrzuceniem wyjątku `OutOfMemoryError`
d) poprawna kompilacja i wykonanie programu

29. Jaki będzie wynik kompilacji i wykonanie poniższego kodu dla standardowo skonfigurowanej wirtualnej maszyny Java?

```
public class A {
   public static void main(String argv[]) {
    double[][] b = new double[10][];
    double c = b[0][0];}
}
```

a) błąd kompilacji
b) poprawna kompilacja, wykonanie zakończone wyrzuceniem wyjątku `NullPointerException`
c) poprawna kompilacja, wykonanie zakończone wyrzuceniem wyjątku `OutOfMemoryError`
d) poprawna kompilacja i wykonanie programu

30. Jaki będzie wynik kompilacji i wykonania poniższego programu?

```
public class B {
    public void m() {
    System.out.print("m() w B");
    }
    public class A {
        public void m() {
        System.out.print("m() w A");
        }
    }
    public static void main(String[] args) {
        new B().new A().m();
    }
}
```

a) błąd kompilacji
b) błąd wykonania
c) `m() w B`
d) `m() w A`