IMIĘ i NAZWISKO:

EGZAMIN „ZEROWY" (14 CZERWCA 2013) – JĘZYK C++

1. Niech wszystkie zmienne poniżej są typu int. Przejdź przez kolejne linie kodu i napisz, jaka będzie na końcu wartość zmiennej e

```
int a = 3; int b = 4; int c = (a % b) * 2; int d = c / b ; int e
= (a + b + c + d) / 4;
cout << e <<endl;
```

2. Napisz wyrażenie, które do strumienia standardowego cout wyśle:
   a) znak końca linii
   b) znak tabulacji
   c) znak " (podwójnego cudzysłowu)
   d) znak prawego ukośnika /

```
cout <<
```

3. Przyjrzyj się poniższemu fragmentowi kodu i dla każdej linii wpisz wartość zmiennych a, b, c po wykonaniu danej linii. Jeśli któraś wartość jest niezdefiniowana, napisz N

```
int a, b, c;
a = 1; b = 2;
a = (b++) + 3;
c = 2*a + (++b);
b = 2*(++c) - (++a);
```

a	b	c

4. W pewny kodzie pojawiła się dla obiektu a typu T taka składnia:

```
a << cout;
```

Kod się kompiluje i wykonuje poprawnie. Proszę wyjaśnić jak i kiedy to jest możliwe.

5. Niech następujące zmienne mają takie wartości:

```
int x = 10, y = 15, z = 20;
```

Proszę podać po kolei wartości logiczne (true, false) poniższych wyrażeń:
a) !(x>10)
b) x<=5 || y<15
c) (x!=5)&&(y!=z)
d) x>=z || (x+y>=z)

6. Kiedy pętla while się zatrzyma (gdy ch przejdzie poza jaką wartość)?

```
char ch = 'D';
while('A' <= ch && ch <= 'Z'){ ch = static_cast<char>( static_cast<int>(ch) + 1); }
```

7. W poniższym fragmencie kodu zakreśl wyraźnie, co jest błędem składniowym (również te miejsca gdzie czegoś ewentualnie brakuje.

```
#include "iostream"
int main() {
  cout << "tekst";
  /*
  // /*** następnie większa ilość kodu ***/
  */
  char c = 32;
}
```

8. W nowym standardzie można użyć using w miejsce typedef. Proszę zatem zapisać z użyciem using poniższy typedef

```
class Foo;
typedef const char& (Foo::*fun) ( int );
```

9. Wśród poniższych wypowiedzi wskaż fałszywą:
   a) namespace definiuje się w przestrzeni globalnej
   b) namespace można zagnieździć wewnątrz innego namespace
   c) namespace można zagnieździć wewnątrz main()
   d) to samo namespace można definiować w wielu plikach
   e) nienazwane namespace służy łączenia wewnętrznego jego zawartości

10. Wśród poniższych zakreś typu niepoprawne w C++
    a) long long
    b) long double
    c) short
    d) unsigned float
    e) unsigned long

11. Mamy tablicę int tab[] { 1, 2, 3, 4, 5 }; Napisz pętlę "for" po całym zakresie tej tablicy (chodzi o odmianę for z nowego standardu C++11) taką, że instrukcja wykonana w pętli będzie przemnażać kolejne wartości przez 2.

12. Dla silnego typu wyliczeniowego (z C++11) jak poniżej, proszę zaznaczyć wszystkie błędne operacje:

```
enum class Figura { kwadrat, kolo, trojkat };
Figura a = 1;
int b = Figura::kolo;
Figura c = trojkat;
```

13. Proszę napisać rzutowanie (w stylu C++) dla wskaźnika ptr takie, że wskaźnik ptr będzie wskaźnikiem stałym do typu T, a chcemy go zrzutować na wskaźnik do stałego obiektu typu T

14. Zakreślić wszystkie zapisy, które są (w C++11) niepoprawne:
    a) T&*
    b) T*&
    c) T&&
    d) T&&*
    e) T**&

15. Jaki błąd popełniono w poniższym kodzie:

```
int& fun(int i) {
 int a = { 0
}; a *= i;
return a;
}
```

16. Co się stanie w poniższym kodzie (zakładamy oczywiście że wszystko co trzeba jest zdefiniowane):

```
class Foo {
 void fun( int );
int fun( char );
public:
  Foo( int );
  float fun( float );
};
int main() {
  Foo(1).fun(1);
}
```

17. Napisz dla klasy Foo definicję operatora przypisania przenoszącego. Nie chodzi o szczegóły, chodzi o o jak wygląda definicja, czym się zaczyna i czym kończy.

18. Dla klasy Foo, która ma po kolei takie składowe:
    int a; std::string s; Bar b; (Bar to jakiś inny typ złożony, który ma konstruktor z parametrem int) napisz konstruktor taki, żeby inicjalizował te pola.

19. Dla klasy Foo, która ma składową std::string s; napisz operator konwersji do typu const char*

20. W poniższym kodzie ktoś zdefiniował już konstruktor przenoszący. Jak zapisać, żeby mimo wszystko (jeśli będzie to możliwe) wygenerował się automatyczny konstruktor kopiujący?

```
class Foo { public:
      Foo( Foo&& );
};
```

21. Których operatorów nie można przeciążyć w C++ (wymień cztery, te inne niż sizeof czy operacje rzutowania). Podpisz je też słownie.

21. Zdefiniuj i zainicjalizuj wskaźnik na metodę składową fun w klasie Foo (pamiętaj, że być może po drodze potrzebne będzie użycie typedef żeby potem można było zainicjalizować a nie przypisać)

```
class Foo { public:
  char& fun( string& );
};
```

22. Jak formalnie wyglądać musi funkcja, którą można podstawić do obsługi wyczerpania się pamięci przy alokowaniu operatorem new. Napisz taką funkcję (jakąś prostą) i ustaw w miejsce domyślnie wywoływanej.

24. Gdzie znajdą się po wykonaniu dziedziczenia w klasie potomnej Derived, metody składowe a( ), b( ), c( ) jeśli:

```
class Base {
  protected: int a( );
  protected: int b( );
  public: int a(int);
  int c( ) const;
};
class Derived : Base { public:
  using Base::a;
};
```

25. Co zostanie wypisane na ekranie:

```
struct A { virtual void print() const { cout<<"Klasa A"<<endl; } };
struct B : A { virtual void print() { cout<<"Klasa B"<<endl; } }; struct
C : B { virtual void print() const { cout<<"Klasa C"<<endl; } }; // w
programie taki kod:
A *ptr1 = new B;
A *ptr2 = new C;
ptr1->print();
ptr2->print();
```

26. W jakiej kolejności zostaną wywołane konstruktory przy tworzeniu obiektu C

```
c; struct A {
  A( ) { cout<<"Klasa A"<<endl; }
};
struct B : virtual A {
  B( ) { cout<<"Klasa B"<<endl; }
};
struct C : B, virtual A {
  A a;
  C( ) { cout<<"Klasa C"<<endl; }
};
```

27. Jest w klasie Foo tablica tab. Napisz jak można ją zainicjalizować dowolnymi różnymi wartościami całkowitymi (składnia C++11)

```
class Foo {
  int tab[3];
public: // tu napisz,



};
```

28. Które z poniżej utworzonych obiektów zgłosza błąd kompilacji (C++11): class Foo { public:

```
 explicit Foo( char ) {}
};
auto main() -> int {
  Foo a('a');
  Foo b{'a'};
  Foo c = 'a';
  Foo d = {'a'};
  Foo e{"a"};
}
```

29. Który konstruktor zostanie wywołany? class Foo { public:

```
  Foo(initializer_list<short>); // 1
  Foo(int); // 2
  Foo(char); // 3
```

```
};
Foo a{ 32 };
```

30. W jaki sposób w klasie potomnej B odziedziczyć konstruktor z klasy A ?

```
class A { public:
   A(int) {}
   A(int, int) {}
};
class B : public A {
   public:
};
```

Napisz też, czy ta się tylko jeden z nich (wybrany), jeśli tak, to jak?

31. Wymień kontenery sekwencyjne w C++11 (jest ich 5) oraz napisz jeden przykład z dowolnym z nich, jak się go tworzy i inicjalizuje (w nowy sposób).

32. Do czego służy decltype, napisz jakiś prosty przykład użycia.

33. Jeśli m to obiekt typu std::map<int, string> to jakiego typu będzie obiekt r ?

```
auto r = m.cbegin( );
```

34. Napisz wyrażenie lambda wewnątrz algorytmu for_each, którym przejedziesz po całym zakresie jakiegoś wektora i poprzez wyrażenie lambda wypiszesz zawartość (elementy) wektora na ekran.

35. Jak zapisuje się w wyrażeniu lambda typ zwracany i kiedy konieczna jest jego specyfikacja (a kiedy nie)?