Příjmení a jméno: Číslo studenta:

Protokol k vyplnění

1. a b c d e ☐☐☐☐☐
2. a b c d e ☐☐☐☐☐
3. a b c d e ☐☐☐☐☐
4. a b c d e ☐☐☐☐☐
5. a b c d e ☐☐☐☐☐
6. a b c d e ☐☐☐☐☐
7. a b c d e ☐☐☐☐☐
8. a b c d e ☐☐☐☐☐
9. a b c d e ☐☐☐☐☐
10. a b c d e ☐☐☐☐☐

1 Metoda
`is_open();`

a) Kontroluje zda je proud připojen k otevřenému souboru
b) Kontroluje zda existuje soubor zadaného jména
c) Slouží k uzavření souboru
d) Slouží k otevření souboru
e) Neexistuje

---

2 Mějme následující prototyp funkce. Je v pořádku? Pokud ne, proč?
`int fce(int cislo, int max = cislo);`

a) Je chybný - za prototypem funkce nesmí být středník
b) Nelze posoudit, bez dalších informací o této funkci
c) Je chybný - není definována standardní hodnota argumentu cislo
d) Je chybný - standardní argument nesmí být lokální proměnná nebo argument
e) Je v pořádku

---

3 Obsluha výjimek je používá:

a) Klíčové slovo generic
b) Klíčová slova stdin, stdout, stderr
c) Klíčová slova try, catch, throw
d) Klíčová slova inline, friend, template
e) Klíčová slova public, private, protected

---

4 Určete jaký bude výstup po provedení následující části zdrojového textu:

```
class base {
 int x;
public:
 void setx(int n) { x = n; }
 void showx() { cout << x; }
};
class derived : private base {
 int y;
public:
 void setxy(int n, int m) { setx(n); y = m; }
 void showxy() { showx(); cout << y; }
};
int main()
{
 derived ob;
 ob.setxy(3, 7);
 ob.showxy();
 ob.setx(10);
 ob.showx();
 return 0;
}
```

a) 3 3 10
b) 7 7 10
c) 7 3 10
d) 3 7 10
e) Zdrojový text obsahuje syntaktické chyby - nelze jej přeložit

---

5 Určete jaký bude výstup po provedení následující části zdrojového textu:

```
class priklad {
 int a, b;
public:
 priklad(int i, int j);  // konstruktor třídy priklad
};
priklad::priklad(int i, int j)
{
 cout << "i + j = " << i+j;
}
int main()
{
 priklad ob;
 return 0;
}
```

a) Zdrojový text obsahuje syntaktické chyby - nelze jej přeložit - nadbytečný středník za poslední složenou závorkou ve třídě priklad

b) Zdrojový text obsahuje syntaktické chyby - nelze jej přeložit - chybí destruktor

c) Zdrojový text obsahuje sémantické chyby - nelze jej přeložit - chybí konstruktor bez argumentů

d) i + j =

e) Nevypíše nic

---

6 Jakým způsobem zajistíme, aby měla funkce přístup k privátním členům třídy bez toho, aby byla jejím členem?

a) Deklarujeme funkci jako friend

b) Každá funkce má přístup k privátním členům třídy i když není členem třídy

c) Deklarujeme funkci jako inline

d) Deklarujeme funkci jako static

e) Takovou funkci nelze vytvořit

---

7 Určete jaký bude výstup po provedení následující části zdrojového textu:

```
int shift_left(int x)
{
 while(x < 4)
  x++;
 return (x << 4);
}
int main()
{
 int a = 1;
 int shl;
 shl = shift_left(a);
 printf("shl = %d", shl);
 return 0;
}
```

a) shl = 256

b) shl = 128

c) shl = 8

d) shl = 64

e) shl = 32

---

8 Určete v jakém pořadí jsou volány konstruktory v následující části zdrojového textu:

```
class A {
public:
 A() { cout << "Konstruktor A, "; }
 ~A() { cout << "Destruktor A, "; }
```

```
};
class B {
public:
 B() { cout << "Konstruktor B, "; }
 ~B() { cout << "Destruktor B, "; }
};
class C : public A, public B {
public:
 C() { cout << "Konstruktor C, "; }
 ~C() { cout << "Destruktor C, "; }
};
int main()
{
 C ob;
 return 0;
}
```

a) Konstruktor B, Konstruktor C, Konstruktor A

b) Konstruktor A, Konstruktor B, Konstruktor C

c) Konstruktor C, Konstruktor B, Konstruktor A

d) Konstruktor A, Konstruktor C, Konstruktor B

e) Konstruktor A, Konstruktor A, Konstruktor B

---

9 Určete jaký bude výstup po provedení následující části zdrojového textu:

```
class base {
public:
 int i;
 base(int x) { i = x; }
 virtual void func() = 0;
};
class derived1 : public base {
public:
 derived1(int x) : base(x) {}
 void func()
 {
  cout << "Derived1: ";
  cout << i * i << " ";
 }
};
class derived2 : public base {
public:
 derived2(int x) : base(x) {}
 void func()
 {
```

```
   cout << "Derived2: ";
   cout << i + i << endl;
  }
};
int main()
{
 base *p;
 derived1 d_ob1(10);
 derived2 d_ob2(10);
 p = &d_ob1;
 p->func();
 p = &d_ob2;
 p->func();
 return 0;
}
```

a) Derived1: 100 Derived2: 100
b) Derived2: 20 Derived2: 20
c) Derived1: 10 Derived2: 10
d) Derived1: 100 Derived1: 100
e) Derived1: 100 Derived2: 20

---

10 Určete jaký bude výstup po provedení následující části zdrojového textu:

```
int i = 0;
for(; i < 10;)
 printf("%d", i++);
```

a) 13579
b) 02468
c) 0123456789
d) Žádný - vznikne nekonečný cyklus
e) 123456789
