Datum: Jméno a Příjmení:.....

1. Uvažte následující fragment programu. Co bude na standardním výstupu po jeho provedení? [3B]

```
int i = 0;
int& ri = i;
int* pi = &ri;
i = 2;
*pi = ri;

std::cout << i << ' ' << ri << ' ' << *pi << '\n';
```

Co bude na výstupu po jeho provedení?

Řešení:

1b za 2
1b za 2
1b za 2

2. Uvažte následující fragment kódu. Co bude na výstupu? [6B]

```
struct object {
    object() { std::cout << "1\n"; }
    object(const object&) { std::cout << "2\n"; }
    object& operator=(const object&) { std::cout << "3\n";
        return *this;
    }
    object(object&&) { std::cout << "4\n"; }
    object& operator=(object&&) { std::cout << "5\n";
        return *this;
    }
    ~object() { std::cout << "6\n"; }
    void pozdrav() {
        std::cout << "Ahoj!\n";
    }
    void pozdrav() const {
        std::cout << "Nazdar!\n";
    }
};

void foo(const object& o){
    o.pozdrav();
}

int main(){
    object o1;
    object o2(o1);
```

```
        o1.pozdrav();
        foo(o1);
        o1 = o1;
    }
```

Řešení:

1b za "1 2"
1b za "Ahoj!"
1b za "2"
1b za "Nazdar!"
1b za "6 3"
1b za "6 6"

3. Je takováto deklarace třídy v pořádku? Proč ne? [2B]

```
class foo {
    int sz;
    int* p;
public:
    foo(int size):sz(size), p(new int[sz]){}
    foo(const foo& rhs) = default;
    foo& operator=(const foo& rhs) = default;
    ~foo(){delete p;}
    …
    …
};
```

Řešení:

0b za "NE!"
2b za "při volání kopírujícího konstruktoru/přiřazení dojde ke zkopírování ukazatele místo dat", "pokud použijeme kopírující konstruktor/přiřazení, dojde k opakovanému smazání pole P" apod.

4. Popište slovy, co znamenají následující deklarace [3B]:

```
using t1 = bool (*) (const std::string&, const std::string&);
using t2 = int*; using ct2 = const t2;
```

Řešení:

1b za "ukazatel na funkci se dvěma parametry typu řetězec a návratovou hodnotou bool"
1b za "ukazatel na int"
1b za "konstantní ukazatel na int"

5. Mějme následující deklarace:

```
double exp(double);
size_t len(std::string&);
```

```
int foo(double, double);
double foo(double, double);
```

Která z následujících volání se nepřeloží? A proč? [5B]

```
a) exp(0);
b) len("idkfa");
c) foo(2, 3);
```

Řešení:

1b za "exp se přeloží"
1b za "len se nepřeloží, ..."
1b za "..., protože nelze převzít dočasný objekt referencí" <--- tohle není snadný bod
"..., protože std::string& nepřijme std::string&&" popřemýšlím, jak tohle zjednodušit
apod.
1b za "foo se nepřeloží, ..."
1b za "..., protože nemohou existovat dvě přetížení lišící se pouze typem návratové hodnoty"
"..., protože není jednoznačně určeno, kterou variantu FOO zavolat"
apod.

6. Mějme tento kód: [1B]

```
float surfaceArea(float x, float y, float z) {
    return 2 * ((x*y) + (x*z) + (y*z));
}
```

Zvolte:

a) Jedná se o definici.
b) Jedná se o deklaraci.

Řešení:

0b za "není to definice"
1b ve všech ostatních případech