Je správně následující deklarace:
static const float x = 10;

Pravda

Jaký je rozdíl mezi strukturou a třídou?

Ve třídě jsou položky implicitně privátní, ve struktuře veřejné.

Kompozice je speciální případ agregace.

Pravda

Objekty jsou vytvářeny pomocí konstruktorů?

Pravda

Jako polymorfismus označujeme fakt, že stejné metody mohou znamenat různé věci u různých tříd.

Pravda

Stav objektu je dán:

Stavem jeho atributů

Klíčové slovo static má následující důsledek:

Takto označený objekt existuje po celou dobu výpočtu.

Jakou hodnotu bude mít symbol EX po následující deklaraci výčtového typu:
enum {E1,E2=16,E3,E4=E3+5,EX];

23

Co je chybně v následujících deklaracích?

```
typedef struct Polozka {
  int klic;
  char *hodnota;
} Polozka;
const Polozka TAB[5];
```

Konstantní objekty musí mít přiřazenu počáteční hodnotu

Najděte chyby v následující deklaraci struktury:

```
struct osoba {
  static char *jmeno;
  char *prijmeni;
  extern int vek;
  float plat;
}
```

Klíčové slovo extern udávající paměťovou třídu se může použít jen na celou strukturu

Viditelnost položek ve struktuře nebo třídě lze upřesnit klíčovými slovy:

Private
Public
Protected

Které z následujících deklarací objektů třídy Osoba jsou správně?

Osoba o;
Osoba o{};
Osoba* o = new Osoba;
Osoba* o = new Osoba();
Osoba* o = new Osoba{};

Jak se nazývají proměnné, které nejsou dynamické?

Statické

Třída může mít libovolný počet destruktorů?

Nepravda

Jak se vytvářejí dynamické objekty?

Pomocí operátoru new

Která z následujících deklarací kopírujícího konstruktoru třídy Osoba je správná?

Osoba(const Osoba&);

Struktura nemůže mít žádný konstruktor?

Nepravda

Která z následujících tvrzení jsou pravdivá?

Pokud nedefinujeme destruktor, překladač jej vytvoří automaticky
Destruktor se volá automaticky, nikoliv programově

Třída může mít jen jeden konstruktor?

Nepravda

Kde se vytvářejí dynamické proměnné?

Na haldě ve volné paměti

Inicializace podle seznamu inicializátorů se provádí:

Podle pořadí položek v deklaraci

Statické datové položky třídy jsou součástí každé instance?

Nepravda

Pokud potřebujeme, aby se volání metody přizpůsobilo dynamicky typu objektu, nad kterým se provádí, musíme:

Metodu označit jako virtuální klíčovým slovem virtual

Pomocí operátoru delete lze rušit objekty:

Vytvořené pomocí operátoru new

Chráněná členská data a metody přístupné pouze pro potomky se označují klíčovým slovem:

Protected

Agregace je vztah, kdy jeden objekt tvoří celek, který jako části obsahuje jiné objekty.

Pravda

Statické metody třídy mohou:

Používat jen statické datové položky této třídy,
Používat jen statické metody této třídy

Uvažte následující fragment kódu:
string str;
Je správně zapsána následující hlavička příkazu cyklu, kterým se projdou všechny znaky řetězce str:
for (auto c: str) ... c ...

Pravda

Třída musí mít vždy právě jeden konstruktor.

Nepravda

Implicitní viditelnost položek ve struktuře (struct) je private?

Nepravda

Co je zapotřebí doplnit do následující deklarace metody print, aby se jednalo o čistě virtuální metodu?
virtual void print()

= 0;

Pokud vytvoříme třídu B pomocí veřejné dědičnosti za třídy A:
class B : public A {...}
pak můžeme:

Pro objekty třídy B používat veřejné metody třídy A
Při realizaci metod třídy B používat veřejné metody třídy A

Co to je čistě abstraktní třída?

Obsahuje pouze čistě virtuální metody

Jazyky C a C++ jsou neimperativní jazyky?

Nepravda

Mějme následující deklarace:
double calc(double);
int count(const string &, char);
Která z následujících volání se nepřeloží?
(a) calc(23.4, 55.1);
(b) count("abcda", 'a');
(c) calc(66);

Výraz (b) je správně
Výraz (c) je správně

Jaký je vztah mezi výpočetními problémy a algoritmy?

Pro některé výpočetní problémy neexistuje žádný algoritmus.
Pro každý výpočetní problém může existovat více algoritmů.

Funkce, které předznačíme jako inline, budou:

Dynamické proměnné v C++ vytváříme pomocí:

Operátoru new

Algoritmus je postup, který má následující vlastnosti:

Je konečný (vždy skončí)?
Je deterministický (v každém kroku víme, který krok následuje)?
Je hromadný (umí řešit všechny instance problému)?

Najděte chyby v následujícím kódu:

```
#include<iostream>
#include<string>
int main() {
  std::string str{ "Hello!" };
  for(auto c : str) { std::cout << "[" << c << "]"; }
  std::cout << '\n';
}
```

Je to správný kód

Jakého typu je výsledek výrazu: 'x' + 'y' ?

Int

Které z následujících stylů se používají při řešení problémů?

Strukturovaný
objektově-orientovaný

Jazyk C je podmnožinou jazyka C++?

Nepravda

Běžné aritmetické konverze (usual arithmetic conversions) způsobí:

Jednodušší argument je převeden na typ složitějšího

Jaký vztah mezi algoritmem a programem platí?

Každý algoritmus lze realizovat programem?