

objektově orientované programování

Započetí testu	Čtvrtek, 1. června 2023, 10:00
Stav	Dokončeno
Dokončení testu	Čtvrtek, 1. června 2023, 10:25
Delta pokusu	25 min.
Získaná	23,25 z možných 30,00 (78%)

1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18
19	20	21	22	23	24	25	26	27

28 29 30

```
public:
    void set_d(int d) const
    {
        this->d = d;
    }

    int get_d() const
    {
        return d;
    }
};
```

Je jejím účelem se v jazyce C++ u objektového programování používat klíčové slovo const. Lze i metodu set_d označit klíčovým slovem const?

Vyberte jednu z nabízených možností:

- ☐ Pravda
- ☒ Nepravda

```

class Point {
    private:
        int x;
    public:
        Point(int x) {
            this->x = x;
        }
}

```

```
int get_s1() const
{
    return d;
}

int value() const
{
    return d * 4;
}

void calc(int v)
{
    d = v * 0.35;
}

private:
void set_s1(int d)
{
    this->d = d;
}
};
```

Které prvky třídy patří do "veřejného nastavení"?

- ☐ a. Data(int d), get_s1(), value(), calc(int v),
- ☐ b. Pouze Data(int d), get_s1(), value()
- ☐ c. Data(), protože je implicitně definovaný; Data(int d), get_s1(), value(), calc(int v)
- ☐ d. Pouze metody get_s1() a value()
- ☐ e. Data(), protože je implicitně definovaný; Data(int d), get_s1(), value()
- ☐ f. Pouze metody get_s1() a value(), protože tyto metody jsou označeny klíčovým slovem **const**.
- ☐ g. Data(int d), get_s1(), value(), set_s1(int d)
- ☐ h. Data(), protože je implicitně definovaný; Data(int d), get_s1(), value(), calc(int v), set_s1(int d)

Úloha 4
Možná
Roční 0,75 / 1,00
Úloha s
všeobecnou

Jaké typy konstruktörů existují? Vyberte iracionální možnosti.

- ☐ a. Implicitní konstruktör - pouze bezparametricky
- ☒ b. Bezparametrický konstruktör.
- ☒ c. Explicitní konstruktör - bezparametricky a parametricky
- ☐ d. Explicitní konstruktör - pouze parametricky
- ☒ e. Parametrický konstruktör.
- ☐ f. Explicitní konstruktör - pouze bezparametricky
- ☒ g. Implicitní konstruktör - pouze parametricky
- ☐ h. Implicitní konstruktör - bezparametricky a parametricky

Úloha 5
Možná
Roční 1,00 / 1,00
Úloha s
všeobecnou

Používá se v jazyce C++ pojem "přetížení funkcí" a "přetížení metod"?

Vyberte jednu z nabízených možností

Napravda

V objektivně-orientovaném programování se používají pojmy "třída" a "objekt". Vyberte správné tvrzení

- ☐ a. Třída popisuje objekt a objekt je pak konkrétní příklad třídy.
- ☒ b. Třída je proměnnou objektu.
- ☐ c. Objekt je proměnnou třídy.
- ☒ d. Pojmy "třída" a "objekt" se používají v důslednosti. Třída je rodič a objekt je potomek.

Co platí o metodě konstruktů? Vyberte správná tvrzení.

☒ a. Může mít parametry a může být definována vícekrát (překrývat metody konstruktů).

☐ b. Provádí se pouze(?) v okamžiku vytvoření objektu.

nepřítel	<input type="checkbox"/> c. Má nevratový typ void. <input checked="" type="checkbox"/> d. Provádí se při rušení objektu. <input type="checkbox"/> e. Metoda je pouze jednou a nemá parametry. <input type="checkbox"/> f. Nemá nevratový typ. <input type="checkbox"/> g. Jmenovité se shodují jako třída.
----------	--

Úroveň 8
Hodnota
Bodů: 100 / 100

Je dáno:

```
class Data {
    ...
}
```

```

    private int d;
    public void set_d(int d_new)
    {
        d = d_new;
    }
}

```

```

    }

    int get_u0() const
    {
        return d;
    }
};

Je správně použit příkaz
date date;

?

Vyberte jednu z nabízených možností
```

```

        }
    }
}

// Nopgrade
class Date {
private:
    int d;

public:
    Date()
    {
        set_d(5);
    }
};
```

Je vytvořen objekt `data2` podle následující deklaraace
`data data1[1:1];`

Jaká hodnota je poskytnuta výrazem `data2_gret_0()` ?

- ☒ a. 30
- ☐ b. 10
- ☐ c. 15
- ☐ d. 20

Úroveň
Rozdíl /
100

Třída s
výsledky

Data udaje:

☐ a.

Realizace dědičnosti.

☒ b.

Deklarace objektu.

☐ c.

Definice třídy.

☐ d.

Uvedení příkaz nebo v objektové-orientovaném programovní použít.

Úroveň
11

Jste donek

```

    Read: 100 /
    100
    (" Ousha x
    vjvzlsou

private:
    int d;
public:
    Oupz(int d)
    {
        this->d = d;
    }
    void set_(int d_new)
    {
        d = d_new;
    }
    int get_d() const

```

Úloha 12	Je správně uvedený fragment definice třídy?	Je správně uvedený fragment definice třídy?
Předmět	class Data	
Rozsah 100	{	
100	private:	
Vstupem	int d21;	
Výstupem	int d2;	
	public:	
	Data() : Data(5)	
	{	
	}	

Úloha 13

Historie

Rešit: 100 / 100

Úloha je vyřešena

Agresivita je vztah, kdy jeden objekt tvrdí, cílek, který jako část obsahuje jiné objekty.

Vyberte jednu z nabízených možností

☒ Pravda
 ☐ Nepravda

Úloha 14

Historie

Rešit: 100 / 100

Úloha je vyřešena

Je uvedená definice třídy Data správná (kompatibilní)?

```
class Data {
}
```

```

    }
    }
}

```

☐ a. Na. Privátne metódy `set_d` je volaná v konštruktoru, ktorý je public. Metódu `set_d` ešte takto volat.

☐ b. Ano.

☐ c. Na. Toto trieda nemôže mať takovú privátne metódu.

☐ d. Nie je rozhodnút. Záleží na prekladači jazyka.

```
private:
    void set_d(int d)
    {
        this->d = d;
    }
};

Lze pro jz vytvořený objekt tento třídy měnit jeho stav?

Vybete jednu z nabízených možností
- Pravda
- Nepravda
```

Úroveň 16 Metriky Body: 100 / 100	V objektově-orientovaném světě, jaké známe <u>principy programování</u> ? a. Zapouzření, Dědičnost, Polymorfismus nepatří mezi principy programování. b. Zapouzření, Dědičnost, Polymorfismus c. Zapouzření, Dědičnost a Polymorfismus nepatří mezi principy programování. d. Objektový styl programování není postaven na uvedených principech.
Úroveň 17 Metriky Body: 100 / 0	Pro uvedenou třídu Data: class Data {

```
3  
4  
5 }  
6  
7 Je správný zápis vytvoření objektu data?  
8  
9 data data;  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151  
152  
153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228  
229  
230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
830  
831  
832  
833
```

```

1 int i, kpi = 1;
2 int npi = 41;
3 int l = 30;
4 while (npi >= l)
5 {
6     kpi = 15;
7     while (kpi <= l << " " << npi << endl)
8     {
9         cout << l << " " << npi << endl;
10    }
11    l = kpi;
12    npi = npi - 1;
13 }
14 }
```

Odpověď:

Je dán uvedený nákaz.

Bodů 100 / 100

Vyberte správnou buzení.

- ☐ a. Objekt ufojje obsahuje pouze datové položky a žádné metody.
- ☒ b. Třída se jmenuje ufojje a objekt se jmenuje Data.
- ☐ c. Název třídy není uveden. Objekt se jmenuje ufojje. Data je klíčové slovo.
- ☐ d. Třída se jmenuje Data a objekt se jmenuje ufojje.

[illegible]

```
#include <iostream>

void m(int a, int b)
{
    ;
}

Vybete jednu z nabízených možností:
☒ Pravda
☐ Nepravda
```

```

// ...
// Chyba - výpis chyby
}

set siostr() {
    ofstream siostr("data.txt");
    siostr << "Vstupni text:\n";
    return s;
}
}

```

☐ a. Je to správné.
☐ b. Chyba - soubor nemá určen mód pro zápis "w".
☒ c. Chyba - soubor není uzavřen

<p>Úloha 23</p> <p>Možná</p> <p>Rešit: 100 / 100</p> <p>1. Úloha z výpočtů</p>	<p>Má lambda funkce definované předem?</p> <p>Vyberte jednu z nabízených možností</p> <p><input type="radio"/> Pravda</p> <p><input checked="" type="radio"/> Nepravda</p>
<p>Úloha 24</p> <p>Možná</p>	<p>Je dána tato funkce/metoda:</p> <pre>void m(int a, int b, int c = 4, int d = 5)</pre>

```

1 // funkce a
2 // volání funkce
3
4 #include <iostream>
5 using namespace std;
6
7 // funkce
8 int a = 10;
9
10 // funkce
11 int a = 10;
12
13 // funkce
14 int a = 10;
15
16 // funkce
17 int a = 10;
18
19 // funkce
20 int a = 10;
21
22 // funkce
23 int a = 10;
24
25 // funkce
26 int a = 10;
27
28 // funkce
29 int a = 10;
30
31 // funkce
32 int a = 10;
33
34 // funkce
35 int a = 10;
36
37 // funkce
38 int a = 10;
39
40 // funkce
41 int a = 10;
42
43 // funkce
44 int a = 10;
45
46 // funkce
47 int a = 10;
48
49 // funkce
50 int a = 10;
51
52 // funkce
53 int a = 10;
54
55 // funkce
56 int a = 10;
57
58 // funkce
59 int a = 10;
60
61 // funkce
62 int a = 10;
63
64 // funkce
65 int a = 10;
66
67 // funkce
68 int a = 10;
69
70 // funkce
71 int a = 10;
72
73 // funkce
74 int a = 10;
75
76 // funkce
77 int a = 10;
78
79 // funkce
80 int a = 10;
81
82 // funkce
83 int a = 10;
84
85 // funkce
86 int a = 10;
87
88 // funkce
89 int a = 10;
90
91 // funkce
92 int a = 10;
93
94 // funkce
95 int a = 10;
96
97 // funkce
98 int a = 10;
99
100 // funkce
101 int a = 10;
102
103 // funkce
104 int a = 10;
105
106 // funkce
107 int a = 10;
108
109 // funkce
110 int a = 10;
111
112 // funkce
113 int a = 10;
114
115 // funkce
116 int a = 10;
117
118 // funkce
119 int a = 10;
120
121 // funkce
122 int a = 10;
123
124 // funkce
125 int a = 10;
126
127 // funkce
128 int a = 10;
129
130 // funkce
131 int a = 10;
132
133 // funkce
134 int a = 10;
135
136 // funkce
137 int a = 10;
138
139 // funkce
140 int a = 10;
141
142 // funkce
143 int a = 10;
144
145 // funkce
146 int a = 10;
147
148 // funkce
149 int a = 10;
150
151 // funkce
152 int a = 10;
153
154 // funkce
155 int a = 10;
156
157 // funkce
158 int a = 10;
159
160 // funkce
161 int a = 10;
162
163 // funkce
164 int a = 10;
165
166 // funkce
167 int a = 10;
168
169 // funkce
170 int a = 10;
171
172 // funkce
173 int a = 10;
174
175 // funkce
176 int a = 10;
177
178 // funkce
179 int a = 10;
180
181 // funkce
182 int a = 10;
183
184 // funkce
185 int a = 10;
186
187 // funkce
188 int a = 10;
189
190 // funkce
191 int a = 10;
192
193 // funkce
194 int a = 10;
195
196 // funkce
197 int a = 10;
198
199 // funkce
200 int a = 10;
201
202 // funkce
203 int a = 10;
204
205 // funkce
206 int a = 10;
207
208 // funkce
209 int a = 10;
210
211 // funkce
212 int a = 10;
213
214 // funkce
215 int a = 10;
216
217 // funkce
218 int a = 10;
219
220 // funkce
221 int a = 10;
222
223 // funkce
224 int a = 10;
225
226 // funkce
227 int a = 10;
228
229 // funkce
230 int a = 10;
231
232 // funkce
233 int a = 10;
234
235 // funkce
236 int a = 10;
237
238 // funkce
239 int a = 10;
240
241 // funkce
242 int a = 10;
243
244 // funkce
245 int a = 10;
246
247 // funkce
248 int a = 10;
249
250 // funkce
251 int a = 10;
252
253 // funkce
254 int a = 10;
255
256 // funkce
257 int a = 10;
258
259 // funkce
260 int a = 10;
261
262 // funkce
263 int a = 10;
264
265 // funkce
266 int a = 10;
267
268 // funkce
269 int a = 10;
270
271 // funkce
272 int a = 10;
273
274 // funkce
275 int a = 10;
276
277 // funkce
278 int a = 10;
279
280 // funkce
281 int a = 10;
282
283 // funkce
284 int a = 10;
285
286 // funkce
287 int a = 10;
288
289 // funkce
290 int a = 10;
291
292 // funkce
293 int a = 10;
294
295 // funkce
296 int a = 10;
297
298 // funkce
299 int a = 10;
300
301 // funkce
302 int a = 10;
303
304 // funkce
305 int a = 10;
306
307 // funkce
308 int a = 10;
309
310 // funkce
311 int a = 10;
312
313 // funkce
314 int a = 10;
315
316 // funkce
317 int a = 10;
318
319 // funkce
320 int a = 10;
321
322 // funkce
323 int a = 10;
324
325 // funkce
326 int a = 10;
327
328 // funkce
329 int a = 10;
330
331 // funkce
332 int a = 10;
333
334 // funkce
335 int a = 10;
336
337 // funkce
338 int a = 10;
339
340 // funkce
341 int a = 10;
342
343 // funkce
344 int a = 10;
345
346 // funkce
347 int a = 10;
348
349 // funkce
350 int a = 10;
351
352 // funkce
353 int a = 10;
354
355 // funkce
356 int a = 10;
357
358 // funkce
359 int a = 10;
360
361 // funkce
362 int a = 10;
363
364 // funkce
365 int a = 10;
366
367 // funkce
368 int a = 10;
369
370 // funkce
371 int a = 10;
372
373 // funkce
374 int a = 10;
375
376 // funkce
377 int a = 10;
378
379 // funkce
380 int a = 10;
381
382 // funkce
383 int a = 10;
384
385 // funkce
386 int a = 10;
387
388 // funkce
389 int a = 10;
390
391 // funkce
392 int a = 10;
393
394 // funkce
395 int a = 10;
396
397 // funkce
398 int a = 10;
399
400 // funkce
401 int a = 10;
402
403 // funkce
404 int a = 10;
405
406 // funkce
407 int a = 10;
408
409 // funkce
410 int a = 10;
411
412 // funkce
413 int a = 10;
414
415 // funkce
416 int a = 10;
417
418 // funkce
419 int a = 10;
420
421 // funkce
422 int a = 10;
423
424 // funkce
425 int a = 10;
426
427 // funkce
428 int a = 10;
429
430 // funkce
431 int a = 10;
432
433 // funkce
434 int a = 10;
435
436 // funkce
437 int a = 10;
438
439 // funkce
440 int a = 10;
441
442 // funkce
443 int a = 10;
444
445 // funkce
446 int a = 10;
447
448 // funkce
449 int a = 10;
450
451 // funkce
452 int a = 10;
453
454 // funkce
455 int a = 10;
456
457 // funkce
458 int a = 10;
459
460 // funkce
461 int a = 10;
462
463 // funkce
464 int a = 10;
465
466 // funkce
467 int a = 10;
468
469 // funkce
470 int a = 10;
471
472 // funkce
473 int a = 10;
474
475 // funkce
476 int a = 10;
477
478 // funkce
479 int a = 10;
480
481 // funkce
482 int a = 10;
483
484 // funkce
485 int a = 10;
486
487 // funkce
488 int a = 10;
489
490 // funkce
491 int a = 10;
492
493 // funkce
494 int a = 10;
495
496 // funkce
497 int a = 10;
498
499 // funkce
500 int a = 10;
501
502 // funkce
503 int a = 10;
504
505 // funkce
506 int a = 10;
507
508 // funkce
509 int a = 10;
510
511 // funkce
512 int a = 10;
513
514 // funkce
515 int a = 10;
516
517 // funkce
518 int a = 10;
519
520 // funkce
521 int a = 10;
522
523 // funkce
524 int a = 10;
525
526 // funkce
527 int a = 10;
528
529 // funkce
530 int a = 10;
531
532 // funkce
533 int a = 10;
534
535 // funkce
536 int a = 10;
537
538 // funkce
539 int a = 10;
540
541 // funkce
542 int a = 10;
543
544 // funkce
545 int a = 10;
546
547 // funkce
548 int a = 10;
549
550 // funkce
551 int a = 10;
552
553 // funkce
554 int a = 10;
555
556 // funkce
557 int a = 10;
558
559 // funkce
560 int a = 10;
561
562 // funkce
563 int a = 10;
564
565 // funkce
566 int a = 10;
567
568 // funkce
569 int a = 1
```

Naziv: Broj(1-100): Godina: Ocjena: Datum:	<p>Primer:</p> <pre>enum {1,12+16,1,8n-1+1,1x};</pre> <p>a. 22</p> <p>b. 24</p> <p>c. 23</p> <p>d. 25</p>
--	---

Po indeksu piteja prve globalni pitomak platiti.

Úloha 27 název: Bodů: 0.00 / 1.00 Průběžná známka: výpočet:	<p> <input type="checkbox"/> a. Je nutno chybu hned odhlásit. <input type="checkbox"/> b. Je nutno chybu hned odhlásit. <input type="checkbox"/> c. Lze zjistit, kde chyba vznikla. <input checked="" type="checkbox"/> d. Nelze zjistit, kde chyba vznikla. </p> <hr/> <p>Může být v jazyce C++ zapsán tento fragment kódu?</p> <pre>void m() { ; }</pre> <p> <input type="checkbox"/> a. Ano <input checked="" type="checkbox"/> b. Ne </p>
--	---

Úloha 29

Množina

100 / 100

100

Vložte z

výsledku

Funkčovní funkce zavede pomocí klíčového slova "route".
Vyberte jednu z nabízených možností.

Prauda

Naprauda

void tisk_znak(char i, char sep)
{
 cout << sep << i << sep;
}

Úloha 30	<p>Úloha 30</p> <p>Přepište zápis parametrů metody tak, aby se defaultně zobrazovaly znaky z v pomlčkách, např. -a -. Doplňte konvenci zápisu a přepišete pouze červeně zvýrazněná.</p> <p>Např. pro volání <code>tisk_znak(v)</code>.</p>
Počet bodů: 1000 / 1000	<p>Odpověď: <input type="text"/></p>

Úloha 30	<p>Jak se jmenuje instance standardního výstupního proudu v jazyce C++?</p>
Počet bodů: 1000 / 1000	<p>Odpověď: <input type="text" value="std::cout"/></p>