

平成21年度	前 訂 其 月 定 其 月 試 馬 矢 氏 是 真
科目名	J a v a プログラミング I
担当教員	石原真紀夫
実施日付	7 月 2 7 日 (月) 4 時 限 (1 組 ・ 2 組)
持ち込み	許 可 ・ 禁 止
情報工学科	年 組 学籍番号

氏名

- 問 1 次の各文を読み、文の内容が正しい場合は「○」を間違いの場合は「×」を解答欄に答えなさい。【各2計40】
- (1) 論理演算子 &&,||,! は「かつ」、「または」、「～ではない」の演算を行います。
- (2) 論理演算子は演算結果として boolean 型の値を返します。
- (3) 1つの式に複数の論理演算子を使うことはできません。
- (4) switch 文は繰り返し文の1つです。
- (5) for 文は前判定ループです。
- (6) while 文は中判定ループです。
- (7) 前判定ループには for 文と do-while 文があります。
- (8) 後判定ループでは必ず1度は繰り返し処理部が実行されます。
- (9) 前判定ループでは1度も繰り返し処理部が実行されないことがあります。
- (10) break 文を用いると実行中の繰り返し文から抜けることができます。
- (11) continue 文は、繰り返し処理を記述できます。
- (12) 配列は異なる型の複数の変数を管理します。
- (13) 1次元配列の最初の配列要素を指定するには添え字に0を指定します。
- (14) 配列の添え字に 1.5 などの実数を指定することができます。
- (15) 1次元配列の配列変数に.length をつけるとその配列の長さを知ることができます。
- (16) 2次元配列の配列要素を指定するには少なくとも3つの添え字が必要です。
- (17) 配列変数は参照型変数の1つです。
- (18) 基本型変数にはint型やdouble型があります。
- (19) クラス型変数に int 型の数値を代入できます。
- (20) クラス型変数は参照型変数の1つです。

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20

問2 次の論理演算子を用いた式を計算するコードです。このコードを実行した直後に変数 b0 から b7 に入っている値(trueまたはfalse)を解答欄に答えなさい。演算子の優先順位表を裏面に載せています。【各1計8】

```
1: boolean b0, b1, b2, b3, b4, b5, b6, b7;
2: boolean a=true, b=true, c=false;
3: b0 = !a;
4: b1 = a && b;
5: b2 = a || c;
6: b3 = !a || b;
7: b4 = a || !a;
8: b5 = b && !b;
9: b6 = a || b && c;
10: b7 = (a || b) && c;
```

b0	b1	b2	b3
b4	b5	b6	b7

問3 配列に関する以下の設問に答えなさい。

設問1 次のコードは1次元配列に代入されている int 型の値を順番に奇数であるか偶数であるかを判断して出力するコードです。空欄に適切なコードを選択肢から選び、その記号を解答欄に答えなさい。【各2計8】

```
1: int[] ary={4, 2, 7, 5, 6, 8, 9, 3, 2, 1};
2: int i;
3: for( 1 ;i< 2 ; 3 ){
4:     if(ary[ 4 ]%2==0)
5:         System.out.print("偶 ");
6:     else
7:         System.out.print("奇 ");
8: }
```

《画面出力》
偶 偶 奇 奇 偶 偶 奇 奇 偶 奇

《選択肢》

1 の選択肢: A. i=0 B. i=1 C. i=2

2 の選択肢: A. ary.length-1
B. ary.length
C. ary.length+1

3 の選択肢: A. i++ B. i-- C. i-=1

4 の選択肢: A. i B. i+1 C. i+2

1	2	3	4
---	---	---	---

設問2 次の2次元配列の初期化を行うコードです。下の図に数値を入れてこの配列を図示しなさい。さらに配列要素のない場所には「×」を記入しなさい。【各1計12】

int[] ary={{0, 1, 0}, {1, 0}, {0, 1, 0, 1}};				
解答欄	1 列目	2 列目	3 列目	4 列目
1 行目				
2 行目				
3 行目				

問4 繰り返し文に関する以下の設問に答えなさい。

設問1 break 文と continue 文の機能の違いを確認するために次の2つのコードを実行した。それぞれのコードを実行した時の画面出力を答えなさい。【各2計4】

(コードA)

```
1: int i=0;
2: while(i<=5){
3:     i++;
4:     if(i==3)break;
5:     System.out.print(i);
6: }
```

(コードB)

```
1: int i=0;
2: while(i<=5){
3:     i++;
4:     if(i==3)continue;
5:     System.out.print(i);
6: }
```

1 (コードAの画面出力)	2 (コードBの画面出力)
---------------	---------------

設問2 while 文と do while 文の処理の違いを確認するために次の2つのコードを実行した。変数 i の値を 4 から 7 のそれぞれの値で初期化した場合の画面出力を答えなさい。ただし、画面に何も出力されない場合は「×」を記入しなさい。【各2計16】

(コードC)

```
1: int i=初期化の値;
2: while(i<=5){
3:     System.out.println(i);
4:     i++;
5: }
```

(コードD)

```
1: int i=初期化の値;
2: do{
3:     System.out.println(i);
4:     i++;
5: }while(i<=5);
```

解答欄

初期化の値	4	5	6	7
(コードC)				
初期化の値	4	5	6	7
(コードD)				

問5 次に4パターンの画面出力を示す。下の4つのコードのそれぞれがどのパターンを出力するのか解答欄に対応するパターン番号を答えなさい。【各3計12】

パターン1	パターン2	パターン3	パターン4
★ ★★ ★★★ ★★★★ ★★★★★	★★★★★ ★★★★ ★★★ ★★ ★	★★★★★ ★★★★ ★★★ ★★ ★	★ ★★ ★★★ ★★★★ ★★★★★

(コードE)

```
1: for(int i=5; i>=1; i--){
2:     for(int j=1; j<=i; j++)
3:         System.out.print("★");
4:     System.out.println();
5: }
```

(コードF)

```
1: for(int i=1; i<=5; i++){
2:     for(int j=1; j<=i; j++)
3:         System.out.print("★");
4:     System.out.println();
5: }
```

(コードG)

```
1: for(int i=1; i<=5; i++){
2:     for(int j=1; j<=5-i; j++)
3:         System.out.print(" ");
4:     for(int j=1; j<=i; j++)
5:         System.out.print("★");
6:     System.out.println();
7: }
```

(コードH)

```
1: for(int i=5; i>=1; i--){
2:     for(int j=1; j<=5-i; j++)
3:         System.out.print(" ");
4:     for(int j=1; j<=i; j++)
5:         System.out.print("★");
6:     System.out.println();
7: }
```

1 (コードE)	2 (コードF)	3 (コードG)	4 (コードH)
----------	----------	----------	----------

お疲れ様です。

※試験監督の先生方へ
配布の際にこの面は生徒に見えても良いです

主な演算子の優先順位

演算子	名前	結合規則
++	後置インクリメント	左
--	後置デクリメント	左
!	論理否定	右
~	1の補数 (反転)	右
+	プラス	右
-	マイナス	右
++	前置インクリメント	右
--	前置デクリメント	右
()	キャスト	右
*	乗算	左
/	除算	左
%	剰余	左
+	加算 (文字列連結)	左
-	減算	左
<<	左シフト	左
>>	右シフト	左
>>>	符号なし右シフト	左
>	より大きい	左
>=	以上	左
<	未満	左
<=	以下	左
==	等価	左
!=	非等価	左
&	ビット論理積	左
^	ビット排他的論理和	左
	ビット論理和	左
&&	論理積	左
	論理和	左
?:	条件	右
=	代入	右
+=, -= など	複合代入演算	右

↑
同じ優先度
↓

優先度高い

優先度低い