TESTDAILY

AP COMPUTER SCIENCE A

AP计算机科学A 新样题

「2021 版本」

有干货 x 有陪伴 x 有进步

AP Computer Science A

TestDaily荣誉出品

Contents

1 Example Questions in the Digital Testing Application

14 Exam Practice Scoring Materials

24 Answers by TestDaily

Section I: Multiple-Choice Question-Part 1

1. Assume that the boolean variables a and b have been declared and initialized. Consider the following expression.

```
(a&&(b||!a))==(a&&b)
```

Which of the following best describes the conditions under which the expression will evaluate to true?

A. Only when a is true

B. Only when b is true

C. Only when both a and b are true

D. The expression will never evaluate to true

E. The expression will always evaluate to true

2. Consider the following code segment.

```
int a=5;
int b=2;
double c =3.0;
System.out.println(5+a/b*c-1);
```

What is printed when the code segment is executed?

A. `0.666666666666667`

B. `9.0`

C. `10.0`

D. `11.5`

E. `14.0`

@ 2021 TestDaily

3. Consider the `process` method. Assume that each of its two parameters is a string of length two or more.

```
public void process (String w1, String w2)
{
    String str1=w1. substring (0, 2) ;
    String str2 = w2. substring (w2. length () -1) ;
    String result =str2 +str1;
    System. out. println (result. indexof (str2) ) ;
}
```

Which of the following best describes the value printed when process is called?

A. The value `0` is always printed.

B. The value `1` is always printed.

C. The value result. length `()` `-` `1` is printed.

D. A substring containing the last character of `w2` is printed.

E. A substring containing the last two characters of `w2` is printed.

4. Which of the following statements assigns a random integer between `25` and `60`, inclusive, to rn?

A. `int rn = (int) (Math. random () *25) +36;`

B. `int rn = (int) (Math, random () *25) +60;`

C. `int rn= (int) (Math. random () *26) +60;`

D. `int rn= (int) (Math. random () *36) +25;`

E. `int rn = (int) (Math. random () *60) +25;`

@ 2021 TestDaily

5. Consider the following method.

```
public int pick (boolean test, int x, int y)
{
  if (test)
  {
    return x;
  }
  else
  {
    return y;
  }
}
```

What value is returned by the following method call?

```
pick(false, pick(true, 0, 1), pick(true, 6,7))
```

A. 0

B. 1

C. 3

D. 6

E. 7

6. Assume that the boolean variables a, b, c, and d have been declared and initialized. Consider the following expression.

```
! (! ( a&&b ) || (c || !d ) )
```

Which of the following is equivalent to the expression?

A. `( a&&b ) && (!c&&d )`

B. `(a ||b ) && (!c&&d)`

C. `(a && b) || (c|| !d)`

D. `(!a || !b) && (!c &&d)`

E. `!(a && b) && (c || !d)`

@ 2021 TestDaily

7. Consider the following method.

```
public int getTheResult (int n)
{
  int product = 1;
  for (int number = 1; number < n; number++)
  {
    if (number %2==0)
    product *= number;
  }
  return product;
}
```

What value is returned as a result of the call `getTheResult(8)`?

A. `48`

B. `105`

C. `384`

D. `5040`

E. `40320`

@ 2021 TestDaily

8. At a certain high school students receive letter grades based on the following scale.

Integer Score	Letter Grade
93 or above	A
From 84 to 92 inclusive	B
From 75 to 83 inclusive	C
Below 75	F

Which of the following code segments will assign the correct string to grade for a given integer score?

I.

```
if (score >=93)
  grade ="A";
if (score >= 84 && score <=92)
  grade = "B";
if (score >=75 && score <=83)
  grade = "C";
if (score<75)
  grade ="F";
```

II.

```
if (score >=93)
  grade = "A";
if (84 <=score <=92)
  grade ="B";
if (75 <= score <=83)
  grade ="C";
if (score<75)
  grade ="F";
```

III.

```
if (score >=93)
  grade ="A";
else if (score >=84)
  grade ="B";
else if (score >=75)
  grade ="C";
else
  grade = "F";
```

A. II only

@ 2021 TestDaily

B. III only

C. I and II only

D. I and III only

E. I, II, and III

9. Consider the following code segment.

```
for (int k=0; k<20; k =k +2)
{
  if(k%3==1)
  {
    System.out.print(k +" ");
  }
}
```

What is printed as a result of executing the code segment?

A. `4 16`

B. `4 10 16`

C. `0 6 12 18`

D. `1 4 7 10 13 16 19`

E. `0 2 4 6 8 10 12 14 16 18`

@ 2021 TestDaily

10. Consider the following output.

```
1  1  1  1  1
2  2  2  2
3  3  3
4  4
5
```

Which of the following code segments will produce this output?

A.

```
for (int j=1; j<=5; j++)
{
  for (int k=1; k<=5; k++)
  {
    System. out. print (j+" ") ;
  }
  System. out. println ();
}
```

B.

```
for (int j = 1; j <=5;j++)
{
  for (int k =1 ;k <=j; k++)
  {
    System. out. print (j+" ");
  }
  System. out. println () ;
}
```

C.

```
for (int j= 1; j<=5; j++)
{
  for (int k=5; k>=1; k--)
  {
    System. out. print (j+" ") ;
  }
  System. out. println () ;
}
```


@ 2021 TestDaily

D.

```
for (int j= 1; j<=5; j++)
{
  for (int k =5;k >=j; k--)
  {
    System. out. print (j+" ") ;
  }
  System. out. println () ;
}
```

E.

```
for (int j= 1; j<=5;j++)
{
  for (int k = j; k <=5; k++)
  {
    System. out. print (k+" ") ;
  }
  System. out. println () ;
}
```


@ 2021 TestDaily

Section II: Free-Response Question-Part 1

1. This question involves reasoning about a simulation of a frog hopping in a straight line. The frog attempts to hop to a goal within a specified number of hops. The simulation is encapsulated in the following `Frogsimulation` class. You will write two of the methods in this class.

```
public class FrogSimulation
{
  /** Distance, in inches, from the starting position
  * to the goal. */
  private int goalDistance;
  /** Maximum number of hops allowed to reach the goal. */
  private int maxHops;

  /** Constructs a FrogSimulation where dist is the
  * distance, in inches, from the starting
  * position to the goal, and numHops is the maximum
  * number of hops allowed to reach the goal.
  * Precondition: dist > 0; numHops > 0
  */
  public Frogsimulation (int dist, int numHops)
  {
    goalDistance =dist;
    maxHops = numHops;
  }
  /** Returns an integer representing the distance, in inches,
  * to be moved when the frog hops.
  */
  private int hopDistance ()
  {/* implementation not shown */}

  /** Simulates a frog attempting to reach the goal as
  described
  * in part (a) .
  * Returns true if the frog successfully reached or passed the
```


@ 2021 TestDaily

```
  * goal during the simulation;
  *   false otherwise
  */
  public boolean simulate ()
  { /* to be implemented in part (a) */}

  /** Runs num simulations and returns the proportion of
  * simulations in which the frog
  * successfully reached or passed the goal.
  * Precondition: num> 0
  */
  public double runsimulations (int num)
  {/* to be implemented in part (b) */}
}
```

a. Write the `simulate` method, which simulates the frog attempting to hop in a straight line to a goal from the frog's starting position of 0 within a maximum number of hops. The method returns `true` if the frog successfully reached the goal within the maximum number of hops; otherwise, the method returns false.

The `FrogSimulation` class provides a method called `hopDistance` that returns an integer representing the distance (positive or negative) to be moved when the frog hops. A positive distance represents a move toward the goal. A negative distance represents a move away from the goal. The returned distance may vary from call to call. Each time the frog hops, its position is adjusted by the value returned by a call to the `hopDistance` method.

The frog hops until one of the following conditions becomes true:

- The frog has reached or passed the goal.
- The frog has reached a negative position.
- The frog has taken the maximum number of hops without reaching the goal.

The following example shows a declaration of a `FrogSimulation` object for which the goal distance is 24 inches and the maximum number of hops is 5. The table shows some possible outcomes of calling the `simulate` method.

```
FrogSimulation sim = new FrogSimulation (24, 5) ;
```


@ 2021 TestDaily

	Values returned by `hopDistance ()`	Final position of frog	Return value of `sim. simulate ()`
Example 1	`5, 7, -2, 8, 6`	`24`	`true`
Example 2	`6, 7, 6, 6`	`25`	`true1`
Example 3	`6, -6, 31`	`31`	`true`
Example 4	`4, 2, -8`	`-2`	`false`
Example 5	`5, 4, 2, 4, 3`	`18`	`false`

Complete method `simulate` below. You must use `hopDistance` appropriately to receive full credit.

```
/** simulates a frog attempting to reach the goal as described
* in part a.
* Returns true if the frog successfully reached or passed the
* goal during the simulation;
*     false otherwise.
public boolean simulate ()
```

b. Write the `runSimulations` method, which performs a given number of simulations and returns the proportion of simulations in which the frog successfully reached or passed the goal. For example, if the parameter passed to `runsimulations` is `400`, and 100 of the 400 simulate method calls returned `true`, then the `runSimulations` method should return `0.25`. Complete method `runsimulations` below. Assume that simulate works as specified, regardless of what you wrote in **part a**. You must use simulate appropriately to receive full credit.

```
/** Runs num simulations and returns the proportion of
simulations
* in which the frog
* successfully reached or passed the goal.
* Precondition: num > 0
*/
public double runSimulations (int num)
```


@ 2021 TestDaily

2. This question involves the implementation of a fitness tracking system that is represented by the `stepTracker` class. A `stepTracker` object is created with a parameter that defines the minimum number of steps that must be taken for a day to be considered *active*.

The `stepTracker` class provides a constructor and the following methods.

- `addDailysteps`, which accumulates information about steps, in readings taken once per day
- `activeDays`, which returns the number of active days
- `averagesteps`, which returns the average number of steps per day, calculated by dividing the total number of steps taken by the number of days tracked

The following table contains a sample code execution sequence and the corresponding results.

Statements and Expressions	Value Returned (blank if no value)	Comment
`StepTracker tr =new StepTracker(10000);`		Days with at least 10,000 steps are considered active.Assume that the parameter is positive.
`tr.activeDays();`	`0`	No data have been recorded yet.
`tr.averagesteps();`	`0.0`	When no step data have been recorded, the `averagesteps` method returns `0.0`.
`tr.addDailysteps(9000);`		This is too few steps for the day to be considered active.
`tr.addDailysteps(5000);`		This is too few steps for the day to be considered active.
`tr.activeDays();`	`0`	No day had at least 10,000 steps.
`tr.averagesteps();`	`7000.0`	The average number of steps per day is (14000/2).
`tr.addDailySteps (13000);`		This represents an active day.
`tr.activeDays();`	`1`	Of the three days for which step data were entered, one day had at least 10,000 steps.
`tr.averagesSteps();`	`9000.0`	The average number of steps per day is (27000/3).
`tr.addDailysteps(23000);`		This represents an active day.
`tr.addDailysteps(1111);`		This is too few steps for the day

@ 2021 TestDaily

		to be considered active.
`tr.activeDays();`	`2`	Of the five days for which step data were entered,two days had at least 10,000 steps.
`tr.averagesteps();`	`10222.2`	The average number of steps per day is (51111/5).

Write the complete `stepTracker` class, including the constructor and any required instance variables and methods. Your implementation must meet all specifications and conform to the example.

@ 2021 TestDaily

2021 Digital Exam Practice

AP Computer Science A

Scoring Materials for Digital Exam Practice

Please note: the digital exam practice resource was developed for students to complete technology checks, experience the digital platform, and practice answering exam questions, including each type of multiple-choice and free-response question they will encounter on exam day. Be aware that on exam day students will complete four free-response questions while this practice includes only two.

This digital exam practice is not a full-length exam, and it does not represent the complete scope of content and skills that students will see on the actual AP exam. This digital exam practice includes only content that would typically be taught in the first half of the school year, following the unit sequence in the AP Computer Science A Course and Exam Description. For more information on the 2021 Exam format, please visit: apcentral.collegeboard.org/pdf/ap-2021-exam-formats.pdf

AP Exams are scored differently than traditional high school or college exams. When an AP Exam is administered, psychometric analysis determines the score ranges corresponding with each AP Exam score (5, 4, 3, 2, and 1) based on a composite score scale that combines and weights the different exam parts. Earning 40-50% of the available points can result in a score of 3 or better on many AP Exams. However, because the number of points corresponding with each AP Exam score can vary on different exams, students and teachers should not use the results of the digital exam practice to predict performance on the 2021 AP Exam.

© 2021 College Board. Visit the College Board on the web: collegeboard.org
AP Central is the official online home for the AP Program: apcentral.collegeboard.org

Multiple-Choice Answer Key

Multiple-Choice Question	Answer
1	E
2	C
3	A
4	D
5	D
6	A
7	A
8	D
9	B
10	D

Applying the Scoring Criteria

Apply the question scoring criteria first, which always takes precedence. Penalty points can only be deducted in a part of the question that has earned credit via the question rubric. No part of a question (a, b, c) may have a negative point total. A given penalty can be assessed only once for a question, even if it occurs multiple times or in multiple parts of that question. A maximum of 3 penalty points may be assessed per question.

1-Point Penalty

v) Array/collection access confusion (`[] get`)
w) Extraneous code that causes side-effect (e.g., printing to output, incorrect precondition check)
x) Local variables used but none declared
y) Destruction of persistent data (e.g., changing value referenced by parameter)
z) Void method or constructor that returns a value

No Penalty

- Extraneous code with no side-effect (e.g., valid precondition check, no-op)
- Spelling/case discrepancies where there is no ambiguity*
- Local variable not declared provided other variables are declared in some part
- `private` or `public` qualifier on a local variable
- Missing `public` qualifier on class or constructor header
- Keyword used as an identifier
- Common mathematical symbols used for operators (× • ÷ ≤ ≥ <> ≠)
- `[]` vs. `()` vs. `<>`
- `=` instead of `==` and vice versa
- `length/size` confusion for array, `String`, `List`, or `ArrayList`; with or without `( )`
- Extraneous `[]` when referencing entire array
- `[i,j]` instead of `[i][j]`
- Extraneous size in array declaration, e.g., `int[size] nums = new int[size];`
- Missing `;` where structure clearly conveys intent
- Missing `{ }` where indentation clearly conveys intent
- Missing `( )` on parameter-less method or constructor invocations
- Missing `( )` around `if` or `while` conditions

Spelling and case discrepancies for identifiers fall under the "No Penalty" category only if the correction can be **unambiguously inferred from context, for example, "ArayList" instead of "ArrayList". As a counterexample, note that if the code declares `"int G=99, g=0;"`, then uses `"while (G < 10)"` instead of `"while (g < 10)"`, the context does **not** allow for the reader to assume the use of the lower case variable.*

© 2021 College Board

Question 1: Methods and Control Structures — 9 points

Canonical solution

(a) — 5 points

```
public boolean simulate()
{
   int position = 0;

   for (int count = 0; count < maxHops; count++)
   {
      position += hopDistance();

      if (position >= goalDistance)
      {
         return true;
      }
      else if (position < 0)
      {
         return false;
      }
   }
   return false;
}
```

(b) — 4 points

```
public double runSimulations(int num)
{
   int countSuccess = 0;

   for (int count = 0; count < num; count++)
   {
      if(simulate())
      {
         countSuccess++;
      }
   }
   return (double)countSuccess / num;
}
```

© 2021 College Board

(a) `simulate`

	Scoring Criteria	Decision Rules	
1	Calls `hopDistance` and uses returned distance to adjust (or represent) the frog's position	Responses **can** still earn the point even if they use `hopDistance()` as a position, like `hopDistance() < 0` Responses **will not** earn the point if they only use `hopDistance()` as a count, like `hopDistance() < maxHops`	**1 point**
2	Initializes and accumulates the frog's position at most `maxHops` times (*must be in context of a loop*)	Responses **will not** earn the point if they do not use a loop	**1 point**
3	Determines if a distance representing multiple hops is at least `goalDistance`	Responses **can** still earn the point even if they use some number of hops `*` `hopDistance()` as the frog's final position	**1 point**
4	Determines if a distance representing multiple hops is less than starting position		**1 point**
5	Returns `true` if goal ever reached, `false` if goal never reached or position ever less than starting position	Responses **can** still earn the point if they have checks for all three conditions and correct return logic based on those checks, even if a check did not earn a point Responses **will not** earn the point if they • do not check all three conditions • only check for `goalDistance` after the loop • only check for starting position after the loop	**1 point**
		Total for part (a)	**5 points**

© 2021 College Board

(b) `runSimulations`

	Scoring Criteria	Decision Rules	
6	Calls `simulate` the specified number of times (*no bounds errors*)	Responses **can** still earn the point even if they do not use the result of calling `simulate` Responses **will not** earn the point if they do not use a loop	**1 point**
7	Initializes and accumulates a count of `true` results	Responses **will not** earn the point if they • initialize the count inside a loop • do not use a loop	**1 point**
8	Calculates proportion of successful simulations using `double` arithmetic	Responses **can** still earn the point if they perform the correct calculation on an accumulated value, even if there was an error in the accumulation Responses **will not** earn the point if they do not divide by the parameter	**1 point**
9	Returns calculated value	Responses **will not** earn the point if they • calculate values using non-numeric types • return a count of simulations	**1 point**
		Total for part (b)	**4 points**
	Question-specific penalties		
	None		
		Total for question 1	**9 points**

© 2021 College Board

Applying the Scoring Criteria

Apply the question scoring criteria first, which always takes precedence. Penalty points can only be deducted in a part of the question that has earned credit via the question rubric. No part of a question (a, b, c) may have a negative point total. A given penalty can be assessed only once for a question, even if it occurs multiple times or in multiple parts of that question. A maximum of 3 penalty points may be assessed per question.

1-Point Penalty

v) Array/collection access confusion (`[] get`)
w) Extraneous code that causes side-effect (e.g., printing to output, incorrect precondition check)
x) Local variables used but none declared
y) Destruction of persistent data (e.g., changing value referenced by parameter)
z) Void method or constructor that returns a value

No Penalty

- Extraneous code with no side-effect (e.g., valid precondition check, no-op)
- Spelling/case discrepancies where there is no ambiguity*
- Local variable not declared provided other variables are declared in some part
- `private` or `public` qualifier on a local variable
- Missing `public` qualifier on class or constructor header
- Keyword used as an identifier
- Common mathematical symbols used for operators (× • ÷ ≤ ≥ <> ≠)
- `[]` vs. `()` vs. `<>`
- `=` instead of `==` and vice versa
- `length/size` confusion for array, `String`, `List`, or `ArrayList`; with or without `( )`
- Extraneous `[]` when referencing entire array
- `[i,j]` instead of `[i][j]`
- Extraneous size in array declaration, e.g., `int[size] nums = new int[size];`
- Missing `;` where structure clearly conveys intent
- Missing `{ }` where indentation clearly conveys intent
- Missing `( )` on parameter-less method or constructor invocations
- Missing `( )` around `if` or `while` conditions

Spelling and case discrepancies for identifiers fall under the "No Penalty" category only if the correction can be **unambiguously inferred from context, for example, "ArayList" instead of "ArrayList". As a counterexample, note that if the code declares `"int G=99, g=0;"`, then uses `"while (G < 10)"` instead of `"while (g < 10)"`, the context does **not** allow for the reader to assume the use of the lower case variable.*

© 2021 College Board

Question 2: Class Design — 9 points

Canonical solution

9 points

```
public class StepTracker
{
   private int minSteps;
   private int totalSteps;
   private int numDays;
   private int numActiveDays;

   public StepTracker(int threshold)
   {
      minSteps = threshold;
      totalSteps = 0;
      numDays = 0;
      numActiveDays = 0;
   }

   public void addDailySteps(int steps)
   {
      totalSteps += steps;
      numDays++;
      if (steps >= minSteps)
      {
         numActiveDays++;
      }
   }

   public int activeDays()
   {
      return numActiveDays;
   }

   public double averageSteps()
   {
      if (numDays == 0)
      {
         return 0.0;
      }
      else
      {
         return (double) totalSteps / numDays;
      }
   }
}
```

© 2021 College Board

`StepTracker`

	Scoring Criteria	Decision Rules	
1	Declares all appropriate `private` instance variables	Responses **will not** earn the point if they • omit keyword `private` • declare variables outside the class	**1 point**
2	Declares constructor: `public StepTracker (int ____)`	Responses **can** still earn the point even if they omit keyword `public` Responses **will not** earn the point if they declare the constructor `private`	**1 point**
3	Uses parameter and appropriate values to initialize instance variables	Responses **can** still earn the point even if they initialize primitive instance variables to default values when declared Responses **will not** earn the point if they • do not use the parameter to initialize some instance variable • do not declare instance variables • initialize local variables instead of instance variables • assign variables to parameters	**1 point**
4	Declares header: `public void addDailySteps (int ____)`	Responses **can** still earn the point even if they omit keyword `public` Responses **will not** earn the point if they declare the method `private`	**1 point**
5	Identifies active days and increments count in `addDailySteps` method	Responses **can** still earn the point even if they put valid comparison erroneously in some other method Responses **will not** earn the point if they • do not use the parameter as part of the comparison • do not increment a count of active days • do not increment an instance variable • compare parameter to some numeric constant	**1 point**
6	Updates other instance variables appropriately in `addDailySteps` method	Responses **will not** earn the point if they • update another instance variable only on active days • update another instance variable inappropriately • do not update appropriate instance variable • update a local variable	**1 point**

© 2021 College Board

7	Declares and implements `public int activeDays()`	Responses **can** still earn the point if they return appropriate count of active days even if the instance variables were updated improperly in `addDailySteps` or `activeDays` Responses **will not** earn the point if they • declare method `private` • return value that is not the number of active days • do not return a value	**1 point**
8	Declares header: `public double averageSteps()`	Responses **can** still earn the point even if they omit keyword `public` Responses **will not** earn the point if they declare the method `private`	**1 point**
9	Returns calculated `double` average number of steps	Responses **can** still earn the point even if they • maintain instance variables improperly but calculate appropriate average • do not handle the special case where no days are tracked Responses **will not** earn the point if they • use integer division • calculate something other than steps divided by days • do not return a value	**1 point**

Question-specific penalties	
None	
Total for question 2	**9 points**

© 2021 College Board

TestDaily

Section I: Multiple-Choice Question-Part 1

1. 答案：E

解释题目：

假设布尔类型值 a 和 b 均已声明且初始化。请思考如下表达式：

```
(a&&(b||!a))==(a&&b)
```

下列哪个选项正确描述了能够使如上表达式的值为“True”的条件？

考点：

Boolean Expressions 的计算（Unit 3）

解题思路：

如果掌握 Distributive Law，即可用快法解题。Distributive Law 的内容是：

A&&(B||C) = A&&B || A&&C。

因此，原式长等号左侧可以化简为：(a&&b || a&&!a)。a&&!a 永远为 false（因为 a 和!a 必有一个为 false),所以(a&&b || a&&!a)可化简为(a&&b || false)。该式是“或”的逻辑，因此只与 a&&b 有关。所以原式长等号左侧可以化简为：a&&b。原式右侧也是 a&&b，两侧永远相等，故永远为 true。

如果不掌握 Distributive Law，则应当列举所有 a 和 b 的值，并比较原式的最终值，如下。

a	b	b || !a	a && (b || !a)	a && b	(a && (b || !a)) == (a&&b)
T	T	T	T	T	T
T	F	F	F	F	T
F	F	T	F	F	T
F	T	T	F	F	T

根据表格数据，在所有条件下，原式（表格中最右侧一列）都为 True，符合 E 选项描述 “always evaluate to true”，故 E 选项正确。

@ 2021 TestDaily

2. 答案：C

解释题目：

请判断题干中代码运行后的打印值。

考点：

`int` 和 `double` 的计算属性、优先级（Unit 1）

解题思路：

本题需要计算 `print` 语句中的计算结果，需要求算的原式为：`5+a/b*c-1`。根据优先级理论，应先计算原式中的乘、除，后计算加、减法。故应先计算 `a/b*c` 的值。

根据 Java 的计算理论，平级的符号，应从左至右，依次计算。故 `a/b*c` 中，应先计算 `a/b`。因 `a` 和 `b` 都是 `int` 类型值，`a=5`，`b=2`，故 `a/b` 的值为 2。注意此处只关注 `a/b`，只有 `int` 类型值，与 `c` 无关，所以不得保留小数。

接下来便可计算 `2*c`。因为 `c` 是 `double` 类型值，故此处结果为 6.0。需要保留小数。

此时，原式已化简为 `5+6.0-1`，便可计算剩余的加减法。此处 6.0 为 `double` 类型值，故结果也需要为 `double` 类型值。经计算，结果为 `10.0`。

完整的计算步骤如下：

```
5 + a / b * c - 1
= 5 + 5 / 2 * 3.0 - 1
= 5 + 2 * 3.0 - 1
= 5 + 6.0 - 1
= 11.0 - 1
= 10.0
```


@ 2021 TestDaily

3. 答案：A

解释题目：

请参考题干中的 process 方法。其中，两个参数均为长度为 2 及以上的 String 类型值。哪个选项最好地描述了该方法的打印值?

考点：

String 的方法(indexOf 及 substring) (Unit 2)

解题思路：

本题考查了 String 方法的使用，如果忘记了方法的作用，请参考 Quick Reference 表格。

假设 w1="123456", w2="abcdef"。

第一行代码获取了 w1 的第 0 位和第 1 位，存在 str1 里。所以 str1="12"。

第二行代码获取了 w2 的最后一位字符，存储在 str2 里。所以 str2="f"。

第三行代码将 str2 和 str1 合并，存在 result 里。注意，str2 在前，然后是 str1。所以 result="f12"。

第四行代码利用 indexOf 方法打印了 str2 在 result 中第一次出现的 index。第三行代码说明了 str2 是在 result 最开始的位置的（result=str2+str1），故 str2 第一次出现的 index 一定为 0。即，在"f12"中，"f"第一次出现的位置是 0，故返回值一定是 0。

本题需要注意，result 由 str2 和 str1 依次拼成，且 indexOf 中的参数是 str2。本题的最终结果其实与参数 w1、w2、第一行和第二行代码完全无关。

@ 2021 TestDaily

4. 答案：D

解释题目：

下列哪个选项可以将一个 25 至 60（包含 25 和 60）之间的一个随机整数赋值给 rn 变量?

考点：

Random Number (Math Class) (Unit 2)

解题思路：

Math.random()的返回值，是一个大于等于 0.0 且小于 1.0 的 double 类型值。

因此，A、B、C 选项皆可直接排除。假设 Math.random()返回了 0.0，可求这三个选项的最小值分别是 36、60、60，均不符题意。

注意，Math.random()可以为 0.0，但不可以是 1.0，永远小于 1.0。即，a*Math.random()一定小于 a。因此，可得(int)Math.random()*a 的最大值是 a-1。

因此，E 选项也不正确。假设 Math.random()返回了最大值，则 E 选项将给 rn 赋值为 84（此时 Math.Random()*60 = 59.999…，强制转换为 int 后是 59，所以 rn=59+25=84），不符题意。

D 选项是求算该随机数的标准公式。假设 Math.random()返回了最大值，则 D 选项将给 rn 赋值为 60（此时 Math.Random()*36 = 35.999…，强制转换为 int 后是 35，所以 rn=35+25=60）；假设 Math.random()返回了 0.0，可求其赋值为 0+25=25。因此，D 选项生成的数值均在范围之内。

@ 2021 TestDaily

5. 答案：D

考点：

`if` 语句/`Method` 与参数（Unit 3/Unit 5）

解释题目：

请参考题干中的 `pick` 方法。当以下列代码调用该方法时，返回值应为多少？

```
pick(false, pick(true, 0, 1), pick(true, 6,7))
```

解题思路：

首先阅读 `pick` 方法的实现过程。根据 `pick` 方法的内部实现，易得：第一个参数 `test` 为 `true`，返回 `x`；第二个参数 `test` 为 `false`，返回 `y`。

题干中，`pick(false, pick(true, 0, 1), pick(true, 6,7))`的最外层 `pick`，给第一个参数 `test` 传入了 `false`，给第二个参数 `x` 传入了 `pick(true, 0, 1)`，给第三个参数 `y` 传入了 `pick(true, 6,7)`。

因为这里第一个参数值 `test` 为 `false`，故应返回第三个参数值，也就是 `pick(true, 6,7)` 的返回值。

因为 `pick(true, 6,7)`的第一个参数值 `test` 为 `true`，故应返回这里的第二个参数值 `x`，即为 `6`。这也就是整个题干的返回值了。本题中无需计算 `pick(true,0,1)`。

@ 2021 TestDaily

6. 答案：A

考点：

De Morgan Law（Unit3）

解释题目：

假设 a、b、c、d 为四个已声明和初始化的布尔类型值。请问题干中的布尔表达式与选项中哪一个是等效的?

解题思路：

原式中的解析式较长，符合使用 De Morgan Law 的条件。De Morgan Law 的内容是：!(A||B) = !A && !B。

套用该定律的完整解题过程：

```
! (! ( a&&b ) || (c || !d ) )
=     (!!( a&&b ))  &&  !(c || !d )   (运用 De Morgan Law)
=       ( a&&b )  &&  !(c || !d )     (化简)
=       ( a&&b )  &&  (!c && !!d )    (运用 De Morgan Law)
=       ( a&&b )  &&  (!c && d )      (化简)
```

与 A 选项一致。

@ 2021 TestDaily

7. 答案：A

考点：

循环次数判断（Unit 4）

解释题目：

请阅读 `getTheResult` 方法并判断调用 `getTheResult(8)` 的返回值。

解题思路：

本题中参数的值，`n`，为 `8`。故循环的完整数值为：`int number = 1; number < 8; number++`。因此，循环变量 `number` 将会变为如下的几个数值：`1`、`2`、`3`、`4`、`5`、`6`、`7`，共循环 `7` 次。

注意，`number` 不会变成 `8`，因为循环的终止条件为 `number<8`，不包含 `8`。

接下来对 `7` 次循环进行分析：

- 当 `number=1` 时，`number % 2` 不等于 `0`，不执行 `if` 内代码。
- 当 `number=2` 时，`number % 2` 等于 `0`，`product = 1*2 = 2`。
- 当 `number=3` 时，`number % 2` 不等于 `0`，不执行 `if` 内代码。
- 当 `number=4` 时，`number % 2` 等于 `0`，`product = 2*4 = 8`。
- 当 `number=5` 时，`number % 2` 不等于 `0`，不执行 `if` 内代码。
- 当 `number=6` 时，`number % 2` 等于 `0`，`product = 8*6 = 48`。
- 当 `number=7` 时，`number % 2` 不等于 `0`，不执行 `if` 内代码。

循环结束，`product` 的值为 48。

@ 2021 TestDaily

8. 答案：D

考点：

if 和 Boolean Expression（Unit 3）

解释题目：

学生的分数转换标准如下：93 及 93+为 A；84~92 为 B；75~83 为 C，75（不含）及以下为 F。三段代码中，哪一个可以依据学生成绩，给出正确的字母转换？

解题思路：

I 是正确的代码。每一个 if 中，都规定了上下限。如 84~92 之间，if 中写为 score>=84 && score<=92。此处可以正确地将"B"分配给 grade。经过代入，也可发现其中没有错误。

II 是错误的。不得写作 84 <=score <=92，这是语法错误。必须写作 score>=84 && score<=92。

III 是正确的。假设 score 为 80，则会进入第二个 else if 中，赋值为 C，正确。代入其他数值，也是正确的。

@ 2021 TestDaily

9. 答案：B

考点：

`for` 循环（Unit 4）

解释题目：

请判断代码的打印值。

解题思路：

本题的循环变量是：`int k=0; k<20; k =k +2`。请注意，k 的变化是 k=k+2，每次均+2。因此，`k` 的变化范围为：0、2、4、6、8、10、12、14、16、18。

根据代码，当 `k%3` 为 `1` 时，应当输出 `k`。

- 0%3 为 0。
- 2%3 为 2。
- 4%3 为 1，输出 4。
- 6%3 为 0。
- 8%3 为 2。
- 10%3 为 1，输出 10。
- 12%3 为 0。
- 14%3 为 2。
- 16%3 为 1，输出 16。
- 18%3 为 0。

故输出为 4 10 16。

@ 2021 TestDaily

10. 答案：D

考点：

`for` 循环（Unit 4）

解释题目：

以下哪段代码可以打印出如上的输出?

解题思路：

依次判断各段代码的输出。

A 选项外层循环 5 次，内层循环 5 次，一共打印了 25 次，不符题意，错误。

B 选项在外层循环第一次的时候，内层循环 1 次（k=1; k<=1; k++），所以第一行的"1"只会打印一个。而题干中的 1 有 5 个，不符题意，错误。

C 选项外层循环 5 次，内层循环 5 次，一共打印了 25 次，不符题意，错误。

D 选项正确。j 的循环变化为 j=1、2、3、4、5。j=1 时，k 的循环共执行 5 遍，所以会输出 5 个 1。j=2 时，k 的循环共执行 4 遍，所以会输出 4 个 2，以此类推。因此，执行后会发现其输出与题干一致。

E 选项错误。打印时，这里输出的值是 k，而非 j，所以打印的第一行会是 1 2 3 4 5，不符题意，错误。

@ 2021 TestDaily

Section II: Free-Response Questions-Part 1

1.

a) 答案：

```
public boolean simulate()
{
int position = 0;
for (int count = 0; count < maxHops; count++)
{
position += hopDistance();
if (position >= goalDistance)
{
return true;
}
else if (position < 0)
{
return false;
}
}
return false;
}
```

解释题目：

编写 `simulate` 方法，以判断 `Frog` 能否于最多跳跃次数内，达到 `goal` 的位置。如果可以，应返回 `true`；反之，返回 `false`。`hopDistance` 方法会返回单次跳跃时 `Frog` 前进的步数。返回值正数为趋近 `goal`，负数为远离 `goal`。每次调用 `hopDistance` 方法后，应当调整 `Frog` 的当前位置。

当以下条件为真时，`Frog` 即停止跳跃：

- `Frog` 达到或超越 `goal`；
- `Frog` 达到了负数位置；
- `Frog` 已经跳跃了最多的次数，但仍未到达 `goal`。

表格是五个例子，以说明各种情况下的返回值。

@ 2021 TestDaily

解题思路：

本题应仔细审题。在本题中，`simulate` 方法应多次利用 `hopDistance` 方法，模拟一次 `Frog` 的完整跳跃过程。首先设立的 `position` 变量，是用于记录 `Frog` 当前位置的变量。

循环采用了 `for` 循环，因为已知循环的最多次数是 `maxHops`，所以可以明确 `for` 循环内的内容。

循环主体中，第一行应当调用 `hopDistance` 方法，并更新 `Frog` 的位置。这样可以完成 `Frog` 的单次跳跃。

接下来通过 `if/else if` 组合，分别判断 `Frog` 是否达到或超过 `goal`，以及 `Frog` 是否到达了负数位置。根据题意，满足第一个条件，返回 `true`；满足第二个条件；返回 `false`。

请注意，根据题干，当以上两个条件中，任意一个为真的时候，应当立即停止跳跃，所以 `if/else if` 应当写在循环主体内。这样，当任意一次循环满足条件的时候，即立即停止循环，并停止方法。这个判断不得写于循环外，否则无法实时判断。

提示：`return` 会立即结束方法，不会执行后续代码。

如果循环内没有执行 `return`，则证明已经完成了 `maxHops` 跳跃却仍未满足 `position>=goalDistance`。此时说明，`Frog` 并未达到 `goal`。根据题意，应当返回 `false`。故循环结束后应 `return false`。至此，答案才覆盖了题干中的三条判断条件。

@ 2021 TestDaily

b) 答案：

```
public double runSimulations(int num)
{
int countSuccess = 0;
for (int count = 0; count < num; count++)
{
if(simulate())
{
countSuccess++;
}
}
return (double)countSuccess / num;
}
```

解释题目：

请编写 runSimulations 方法，多次执行指定次的 Frog 跳跃模拟，判断每次模拟中 Frog 是否到达了 goal，并返回成功率。如果参数为 400，即执行 400 次模拟；如果其中 100 次模拟达到了 goal，则应返回 0.25。

解题思路：

本题利用抽象思维，多次调用 a 问编写的方法。首先设立 countSuccess 变量，记录成功次数。接下来，因为循环次数已知（即为参数 num 次），故选择 for 循环，并循环 num 次。每次循环中，应调用 simulate 方法。若 simulate 返回值为 true，则应让 countSuceess 变量自增一次。

循环结束后，应当注意审题。本题要求返回成功的比率，故应当用 countSuccess 除以 num，得到比率。注意，两个变量都是 int，所以应当先将 countSuccess 强制转换为 double 类型，然后相除。此处还可以写作 countSuccess / (double) num。此处不能写作(double)(countSuccess/num)，否则返回值永远为 0.0。

@ 2021 TestDaily

另外，在第一行将 `countSuccess` 变量声明为 `double` 也是可行的，这样无需在最后将 `countSuccess` 强制转换为 `double` 了。

@ 2021 TestDaily

2.

答案:

```
public class StepTracker {
    private int minSteps;
    private int totalSteps;
    private int numDays;
    private int numActiveDays;

    public StepTracker(int threshold) {
        minSteps = threshold;
        totalSteps = 0;
        numDays = 0;
        numActiveDays = 0;
    }

    public void addDailySteps(int steps) {
        totalSteps += steps;
        numDays++;
        if (steps >= minSteps) {
            numActiveDays++;
        }
    }

    public int activeDays() {
        return numActiveDays;
    }

    public double averageSteps() {
        if (numDays == 0) {
            return 0.0;
        } else {
            return (double) totalSteps / numDays;
        }
    }
}
```


@ 2021 TestDaily

解释题目：

本题要求我们写出完整的 `stepTracker` 类。该类创建时需要有一个参数，以定义能够被认为是活跃的最小步数。

本类需要有一个构造方法，以及如下若干方法：

1. `addDailySteps` 方法，用于累计每日的步数信息。

2. `activeDays` 方法，用于返回活跃天数。

3. `averageSteps`，返回日均步数；用总步数除以天数。

表格中为方法调用示例。

解题思路：

书写全类，要依此次序：成员变量、构造器和其他方法。

成员变量：经审题，本题首先需要记录创始时的参数，即多少步数才能算作 `active`。根据方法需求，我们需要记录总天数、总步数以计算平均步数。同时，为了满足 `activeDays` 方法的需求，还需要设立记录 `active` 天数的方法。因此，共需 `4` 个成员变量。成员变量的名称无需和答案一致，但需要符合命名规范，并避免混淆。

构造器：根据题意，需要提供一个构造器，需要接受一个参数（即为 `minSteps`）。因此，需要写接受一个参数的构造器，并将其它成员变量设为初始值 `0`。

其他方法：

1. `addDailySteps` 方法

根据示范，本方法需要接受一个参数，将其加入 `totalSteps`，并让天数加一。完成这两步操作后，需要判断该日的步数是否满足 `active` 标准，并对 `numActiveDays` 变量进行相应操作。此处便可使用 `if` 条件进行判断。本方法无返回值。

@ 2021 TestDaily

2. `activeDays` 方法

本方法实为一个 `Accessor`，只需返回该变量即可。该变量为 `int` 类型，故返回值为 `int`。

3. `averageSteps` 方法

本题需要实现平均步数的计算，其中需要返回 `double` 类型的计算结果。注意本题不得直接将 `totalSteps` 和 `numDays` 相除。为了避免无数据时调用出错，必须通过 `if/else` 结构，为 `numDays==0` 的情况另写一种返回值 `0.0`。只有在 `numDays!=0` 时，才可以返回二者的商。注意，这里需要将 `totalSteps` 强制转换为 `double`，才能计算精准的数值。

本题方法中调用变量时，无需添加 `this.` 前缀，可以省略。但如果构造器的参数与成员变量同名（假设 `threshold` 写作了 `minSteps`），则需要用 `this.` 加以区分。

@ 2021 TestDaily

