SỞ GIÁO DỤC VÀ ĐẠO TÀO

THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG THPT NGUYỄN TẤT THÀNH

**GỢI Ý HƯỚNG DẪN HỌC SINH TỰ HỌC – HỌC KÌ II - TUẦN 06**

**MÔN Tin HỌC – KHỐI 11**

| **NỘI DUNG** | |
| --- | --- |
| **Tên bài học/ chủ đề** | **Chủ đề 4: kiểu dữ liệu CÓ CẤU TRÚC**  **Bài 12. KIỂU xâu (string)** |
| **Hoạt động 1: *Đọc tài liệu và thực hiện các yêu cầu.*** | 1. **Tài liệu tham khảo:**  * Sách giáo khoa Tin học 11 (bản chuẩn):   Bài 12: Kiểu xâu   * Video bài giảng:   Bài 12:  <https://youtu.be/kz9En9_owyc?list=PL0m7CL-Dv3NcxEahiU6P-J48Wj5ukuTUz>   * Tóm tắt kiến thức cần ghi nhớ (phụ lục 1 – đính kèm) * Mục tiêu **bài 12**: **Kiểu xâu (string)** * Về kiến thức: * Nắm được khái niệm kiểu dữ liệu xâu; * Nắm được một số hàm và thủ tục giúp người lập trình xử lí dữ liệu kiểu xâu và dữ liệu văn bản; * Về kĩ năng: * Khai báo kiểu xâu; * So sánh hai xâu; * Nhận biết và bước đầu sử dụng được các hàm và thủ tục chuẩn xử lí dữ liệu kiểu xâu; * Về thái độ: * Giúp học sinh thấy tự tin hơn khi lập chương trình giải các bài toán cần sử dụng kiểu dữ liệu xâu;  1. **Yêu cầu:**  * Học sinh ghi chép cẩn thận phụ lục 1 vào vở bài tập, cần đánh dấu, tô màu các cú pháp (bộ qui tắc để viết chương trình) học sinh thấy khó ghi nhớ, các ví dụ, các chú ý, các trình tự thao tác. Khi đánh dấu, tô màu, kẻ bảng, kẻ khung, gạch dưới, vẽ hình cần cẩn thận, rõ, sạch. * Trong quá trình đọc và ghi chép, nếu thắc mắc học sinh điền vào Phiếu tổng hợp thắc mắc (phụ lục 2 – đính kèm) và sớm liên hệ với giáo viên để được kịp thời giải đáp. |
| **Hoạt động 2: Kiểm tra, đánh giá quá trình tự học.** | * Hoàn thành phiếu học tập (phụ lục 3 – đính kèm), chụp và nộp lại theo yêu cầu của giáo viên. |

**PHỤ LỤC 1**

# **CHƯƠNG IV. KIỂU DỮ LIỆU CÓ CẤU TRÚC**

## BÀI 12. KIỂU XÂU

* Dữ liệu trong các bài toán không chỉ thuộc kiểu số mà cả kiểu phi số - dạng kí tự. Dữ liệu kiểu xâu là dãy các kí tự.

Ví dụ: ‘Sai gon’ ‘2020 la nam Canh Ty’

* Xâu là một dãy các kí tự trong bộ mã ASCII, mỗi kí tự được gọi là một phần tử của xâu. Số lượng kí tự trong một xâu được gọi là **độ dài của xâu**. **Xâu có độ dài bằng 0 gọi là XÂU RỖNG.**
* Các ngôn ngữ lập trình đều có qui tắc, cách thức cho phép xác định:
* Tên kiểu xâu;
* Cách khai báo biến kiểu xâu;
* Số lượng kí tự của xâu;
* Các phép toán thao tác với xâu;
* Các tham chiếu tới phần tử của xâu.
* Có thể coi xâu như mảng một chiều mà mỗi phần tử là một kí tự. Các kí tự của xâu được đánh số thứ tự, thường bắt đầu từ **1** (Pascal qui định phần tử đầu, bắt đầu từ 1).
* Biểu thức gồm các toán hạng là biến xâu, biến kí tự hoặc hằng xâu được gọi là biểu thức xâu.
* Tương tự mảng, tham chiếu tới phần tử của xâu được xác định bởi tên biến xâu và chỉ số đặt trong cặp ngoặc vuông [ và ]

**Ví dụ**, giả sử có biến *hoten* lưu trữ giá trị hằng xâu ‘Nguyen Tat Thanh’ thì *hoten*[6] cho ta kí tự ‘n’ là **kí tự thứ sáu** của biến xâu*hoten*.

1. **Khai báo**

Để khai báo dữ liệu kiểu xâu ta sử dụng từ khóa **string**, tiếp theo là độ dài lớn nhất của xâu (**không vượt quá 255**) được ghi trong cặp ngoặc vuông **[** và **]**

* **Biến kiểu xâu có thể khai báo như sau:**

**var***<tên biến>:* ***string[****độ dài lớn nhất của xâu****];***

**Ví dụ. var** Hoten**:** **string**[26];

Trong mô tả xâu có thể bỏ qua phần khai báo độ dài, chẳng hạn:

var ChuGiai: **string**;

Khi đó, độ dài lớn nhất của xâu sẽ nhận giá trị ngầm định (**default**) là **255**.

1. **Các thao tác xử lí xâu**
2. **Phép ghép xâu**, **kí hiệu là dấu cộng** (**+**), được sử dụng để ghép nhiều xâu thành một. Có thể thực hiện phép ghép xâu đối với các hằng và biến xâu.

**Ví dụ**: ‘Sai’ + ‘Gon’ + ‘ – ‘ + ‘TP.HCM’ cho xâu kết quả là:   
 ‘Sai Gon – TP.HCM’

1. Các **phép so sánh**: =, <>, <, >, <=, >= có thứ tự ưu tiên thực hiện thấp hơn phép ghép xâu và thực hiện việc so sánh hai xâu theo qui tắc sau:

* Xâu A là **lớn hơn** xâu B nếu như kí tự đầu tiên khác nhau giữa chúng kể từ trái sang trong xâu A có mã ASCII **lớn hơn**.
* Nếu A và B là các xâu **có độ dài khác nhau** và A là đoạn đầu của B thì A nhỏ hơn B.

**Ví dụ**: ‘que huong’ < ‘que huong toi’

* Hai xâu được gọi là bằng nhau nếu như chúng giống nhau hoàn toàn.

**Ví dụ**: ‘Sai gon’ = ‘Sai gon’ hoặc ‘Sai Gon’ = ‘Sai Gon’ hoặc   
‘sai gon’ = ‘sai gon’ hoặc ‘SAI GON’ = ‘SAI GON’

* **Để xử lí xâu, có thể sử dụng các thủ tục và hàm chuẩn dưới đây:**

1. **Thủ tục** (procedure): **delete(st, vt, n)** thực hiện việc **xóa** **n** kí tự tại vị trí **vt** của biến xâu **st**.

|  |  |  |
| --- | --- | --- |
| **Giá trị st** | **Thao tác** | **Kết quả** |
| ‘abcdef’ | delete(st, 5, 2) | ‘abcd’ |
| ‘Song Hong’ | deltete(st, 1, 5) | ‘Hong’ |

1. Thủ tục **insert(s1, s2, vt)** chèn xâu **s1** vào xâu **s2** tại vị trí **vt**.

|  |  |  |  |
| --- | --- | --- | --- |
| **Giá trị s1** | **Giá trị s2** | **Thao tác** | **Kết quả** |
| ‘ PC ‘ | ‘IBM486’ | insert(s1, s2, 4) | ‘IBM PC 486’ |
| ‘1’ | ‘Hinh .2’ | insert (s1, s2, 6) | ‘Hình 1.2’ |

1. **Hàm** (function): **copy(s, vt, n)** trả về xâu (**lấy**) gồm **n** kí tự liên tiếp tại vị trí **vt** của xâu **s**

|  |  |  |
| --- | --- | --- |
| **Giá trị s** | **Biểu thức** | **Kết quả** |
| ‘Bai hoc thu 9’ | copy(s, 9, 5) | ‘thu 9’ |

1. Hàm **length(s)** trả về giá trị số là độ dài xâu **s**.

|  |  |  |
| --- | --- | --- |
| **Giá trị s** | **Biểu thức** | **Kết quả** |
| ‘500 ki tu’ | length(s) | 9 |

1. Hàm **pos(s1, s2)** trả về giá trị số chỉ **vị trí xuất hiện đầu tiên** của xâu **s1** trong xâu **s2**.

|  |  |  |
| --- | --- | --- |
| **Giá trị s2** | **Biểu thức** | **Kết quả** |
| ‘abcdef’ | pos(‘cd’, s2) | 3 |
| ‘abcdef’ | pos(‘k’, s2) | 0 |

1. Hàm **UPCASE(ch)** trả về chữ cái in hoa ứng với chữ cái trong **ch**.

|  |  |  |
| --- | --- | --- |
| **Giá trị ch** | **Biểu thức** | **Kết quả** |
| ‘d’ | upcase(ch) | ‘D’ |
| ‘E’ | upcase(ch) | ‘E’ |

🖎 **Lưu ý**:

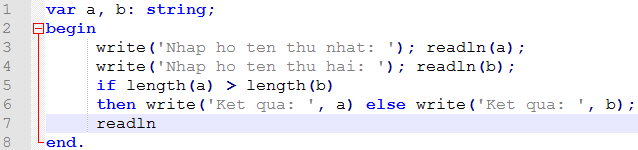
* Xâu được tạo thành bởi các kí tự, trong đó có thể có dấu cách. Dấu cách thể hiện trong các văn bản là phần trống ngăn cách giữa hai từ viết liên tiếp. Kí tự này được gõ bằng phím dài nhất trên bàn phím (**Space bar**);
* Trong chương trình, **khi viết một xâu kí tự, ta phải viết xâu đó giữa hai dấu nháy đơn**. **Nhưng khi nhập từ bàn phím** giá trị một xâu, ta **chỉ cần GÕ các kí tự thuộc xâu đó**, rồi nhấn phím **Enter** (nghĩa là không cần gõ thêm vào cặp dấu nháy đơn, rồi nhấn phím **Enter**).
* Xâu **chỉ gồm một dấu cách** được viết là ‘**␣**’ Ví dụ: b:= ‘**␣**’;
* Để viết **xâu rỗng** ta viết hai dấu nháy đơn liền nhau. Ví dụ **b:=’’;** {để khởi tạo xâu rỗng cho xâu b}
* Khi so sánh hai xâu, xâu có độ dài nhỏ hơn có thể là xâu lớn hơn (>), ví dụ: **‘anh’ < ‘ba’**
* Khi sử dụng lệnh gán, ta có thể gán giá trị là một kí tự cho một biến xâu kí tự nhưng việc gán giá trị là một xâu kí tự cho một biến kiểu kí tự là **KHÔNG HỢP LỆ** dù xâu đó có độ dài bằng 1.
* Tham số của các thủ tục và hàm chuẩn phải hợp lí, chẳng hạn **không thể dùng** **insert(s1, s2, 10) khi length(s2)< 10.**

1. **Một số ví dụ**

***Ví dụ 1***

**Bài toán so sánh hai xâu:**

***“nhập họ tên của hai người vào hai biến xâu và đưa ra màn hình xâu dài hơn, nếu bằng nhau thì đưa ra xâu nhập sau”.***

******

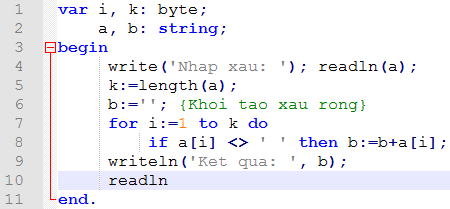
⇨ Khi chạy chương trình, kết quả có thể có dạng như hình dưới đây:

****

***Ví dụ 2***

**Bài toán đưa ra màn hình xâu thu được bằng việc loại bỏ các dấu cách (nếu có) của xâu nhập vào từ bàn phím:**

***“nhập một xâu từ bàn phím và đưa ra màn hình xâu thu được từ nó sau khi loại bỏ dấu cách nếu có”***



⇨ Khi chạy chương trình, kết quả có thể có dạng như hình dưới đây:

****

**PHỤ LỤC 2**

**PHIẾU TỔNG HỢP CÂU HỎI – THẮC MẮC   
CỦA HỌC SINH TRONG QUÁ TRÌNH TỰ HỌC – HỌC KÌ II - TUẦN 06**

Trường THPT Nguyễn Tất Thành

Lớp: 11A…

Họ tên học sinh: Stt:

|  |  |  |
| --- | --- | --- |
| **BÀI** | **NỘI DUNG HỌC TẬP** | **CÂU HỎI CỦA HỌC SINH** |
| 12. Kiểu xâu (string) | Mục: …  Phần: … |  |
|  |  |  |

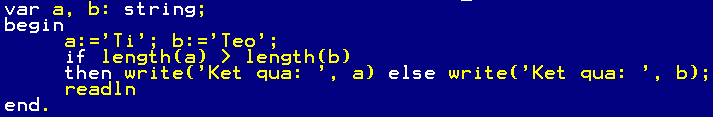
**PHỤ LỤC 3**

**PHIẾU HỌC TẬP**

**Phần 1. Câu hỏi**

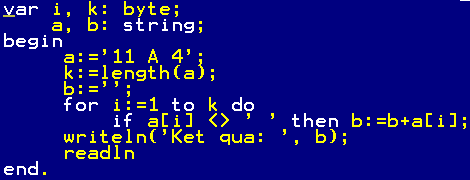
**Bài 12. Kiểu xâu**

1. Xâu là dãy các kí tự trong bộ mã \_\_\_\_\_, mỗi kí tự được gọi là một phần tử của xâu. Số lượng kí tự trong một xâu được gọi là độ dài xâu. Xâu có độ dài bằng 0 gọi là xâu rỗng.
2. ACSI
3. ASCI
4. ACSII
5. ASCII
6. Hãy chọn **xâu rỗng**:
7. ‘’
8. ‘0’
9. ‘xau rong’
10. ‘xâu rỗng’
11. Để **khai báo biến xâu**, ta cần phải có các thông tin gì?
12. Tên biến xâu.
13. Độ dài lớn nhất của xâu.
14. Tên biến xâu và độ dài lớn nhất của xâu.
15. Tất cả đáp án A, B và C đều sai.
16. Để **khai báo biến kiểu xâu**, ta sử dụng cú pháp:
17. var <tên biến>: string
18. var <tên biến>: string[255];
19. var <tên biến>: string[độ dài của xâu];
20. var <tên biến>: string[độ dài lớn nhất của xâu];
21. Khai báo biến **ten** để lưu trữ tên của một học sinh (biết độ dài tên không quá **6** kí tự):
22. var ten: string
23. var ten: string6;
24. var ten: string[6]
25. var ten: string[6];
26. **Độ dài lớn nhất** của xâu có thể được khai báo là bao nhiêu?
27. 255
28. 525
29. 552
30. 555
31. Cho khai báo **var chugiai: string;** Hãy cho biết độ dài lớn nhất của xâu **chugiai** là:
32. 255
33. 525
34. 552
35. 555
36. Để **tham chiếu tới phần tử** của xâu ta sử dụng cú pháp:
37. Tên biến xâu[chỉ số xâu]
38. Tên biến xâu[chỉ số phần tử]
39. Tên biến xâu[kiểu số xâu]
40. Tên biến xâu[kiểu phần tử]
41. Cho khai báo **var b: string;** để khởi tạo xâu rỗng cho biến **b** ta sử dụng lệnh:
42. b := ‘’;
43. b := ‘ ’;
44. b := 0;
45. b := ‘0’;
46. Cho khai báo **var hoten: string[30];** để lưu trữ hằng xâu **‘Nguyen Van Tung’** vào biến **hoten**, thực hiện:
47. readln(hoten);
48. hoten := ‘Nguyen Van Tung’;
49. for i := 1 to 15 do readln(hoten[i]);
50. Tất cả đáp án A, B và C đều đúng.
51. Cho khai báo **var hoten: string[30];** để lưu trữ hằng xâu **‘Huynh Ngoc Lan’** vào biến **hoten**, thực hiện:
52. readln(hoten);
53. hoten := ‘Huynh Ngoc Lan’;
54. for i := 1 to 14 do readln(hoten);
55. Đáp án B và C đúng.
56. Cho khai báo **var hoten: string[30];** để tham chiếu tới phần tử thứ **11** của biến xâu **hoten** ta sử dụng:
57. hoten
58. hoten[10]
59. hoten11
60. hoten[11]
61. Hãy cho biết độ dài của hằng xâu **‘Nguyen Le Huyen’** là bao nhiêu?
62. 13
63. 14
64. 15
65. 16
66. Cho trước **hoten := ‘Nguyen Le Huyen’;**  thì phần tử nào của xâu **hoten** có giá trị là **‘g’**:
67. hoten
68. hoten2
69. hoten[1]
70. hoten[2]
71. Cho trước **hoten := ‘Nguyen Le Huyen’;**  để thay đổi giá trị lưu trữ của biến **hoten** thành **‘Nguyen Le Tuyen’**, ta dùng lệnh:
72. hoten := ‘Nguyen Le Tuyen’;
73. hoten[11] := ‘T’;
74. Cả đáp án A và B đều sai.
75. Cả đáp án A và B đều đúng.
76. Cho trước khai báo **var hoten: string[30];** và 2 phép gán **ho := ‘Tran’; ten := ’Teo’;** để gán giá trị **‘Tran Quang Teo’** cho biến **hoten**, ta dùng lệnh:
77. hoten := ho + ten;
78. hoten := ho + ‘Quang’ + ten;
79. hoten := ho + ‘ Quang ‘ + ten;
80. hoten := ho + “ Quang “ + ten;
81. Cho trước khai báo **var hoten: string[30];** và 2 phép gán **ho := ‘Nguyen ’; ten := ’Tung’;** để gán giá trị **‘Nguyen Thanh Tung’** cho biến **hoten**, ta dùng lệnh:
82. hoten := ho + ten;
83. hoten := ho + ‘Thanh’ + ten;
84. hoten := ho + ‘Thanh ‘ + ten;
85. hoten := ho + ‘ Thanh ‘ + ten;
86. Cho **xâu A := ’114115’**; và **xâu** **B := ’1131150’;** khi đó:
87. xâu A > xâu B
88. xâu A < xâu B
89. xâu A <= xâu B
90. xâu A = xâu B
91. Cho **xâu A := ’Tin hoc’**; và **xâu** **B := ’Tin hoc 11’;** khi đó:
92. xâu A < xâu B
93. xâu A > xâu B
94. xâu A >= xâu B
95. xâu A = xâu B
96. Thủ tục **delete(st, vt, n)** có công dụng:
97. Xóa *n* kí tự.
98. Xóa *n* kí tự của biến xâu *st*.
99. Xóa *n* kí tự của biến xâu *st* bắt đầu từ vị trí *vt*.
100. Tất cả đáp án A, B và C đều sai.
101. Sau khi thực hiện các lệnh **st := ’Computer Hoang Long’; delete(st, 9, 6);** thì biến **st** sẽ lưu trữ giá trị:
102. ’ComputerLong’
103. ’Computer Long’
104. ’CompuLong’
105. ’Compu Long’
106. Hàm **Length(s)** có công dụng:
107. Cho giá trị số.
108. Cho giá trị xâu.
109. Cho giá trị kí tự.
110. Cho giá trị là độ dài xâu *s*.
111. Cho trước **s := ‘500 ki tu’;** khi đó biểu thức **Length(s)** sẽ có giá trị là:
112. 9
113. 10
114. 11
115. 12
116. Cho trước **hoten := ‘Nguyen Le Huyen’;**  để tham chiếu đến kí tự cuối cùng của biến xâu **hoten** ta sử dụng:
117. hoten[15]
118. hoten[length(hoten)]
119. Cả đáp án A và B đều sai.
120. Cả đáp án A và B đều đúng.
121. Kết quả của biểu thức **Length(’Ba muoi bay’)** là:
122. Ba muoi bay
123. ‘Ba muoi bay’
124. 37
125. 11
126. Cho trước **st := ‘Computer Hoang Long’;** Hãychọn tham số thích hợp cho thủ tục **delete(st,\_\_\_,\_\_\_);** để khi thực hiện, biến **st** sẽ có giá trị là **‘Hoang Long’**:
127. 1, 8
128. 8, 1
129. 1, 9
130. 9, 1
131. Thủ tục **insert(s1, s2, vt)** có công dụng:
132. Chèn xâu *s1*.
133. Chèn xâu *s1* vào xâu *s2*.
134. Chèn xâu *s1* vào xâu *s2* bắt đầu ở vị trí *vt*.
135. Tất cả đáp án A, B và C đều sai.
136. Sau khi thực hiện các lệnh **hoten := ’Mai Hoa’;** **insert(’Ngoc’, hoten, 4);** thì biến **hoten** sẽ lưu trữ giá trị:
137. ’MaiNgocHoa’
138. ’Mai NgocHoa’
139. ’MaiNgoc Hoa’
140. ’Mai Ngoc Hoa’
141. Cho trước **ht:= ’Tran Hung’;** Hãy chọn tham số thích hợp cho thủ tục **insert(\_\_\_,ht,\_\_);** để khi thực hiện, biến **ht** sẽ nhận giá trị **’Tran Thanh Hung’**:
142. ‘Thanh’, 5
143. ‘Thanh ’, 6
144. ‘ Thanh ’, 5
145. ‘ Thanh ’, 6
146. Hàm **copy(S, vt, N)** có công dụng:
147. Tạo xâu gồm *N* kí tự liên tiếp.
148. Tạo xâu gồm *N* kí tự liên tiếp bắt đầu từ vị trí *vt*.
149. Tạo xâu gồm *N* kí tự liên tiếp bắt đầu từ vị trí *vt* của xâu *S*.
150. Tất cả đáp án A, B và C đều sai.
151. Cho trước **mahocsinh := ’11A10.23’;** thì biểu thức **copy(mahocsinh, 1, 5)** có kết quả:
152. ’0’
153. ’11A10’
154. ’10.23’
155. ’11A10.23’
156. Cho trước **hoten := ’Nguyen Thi Ngoc’;** Hãy chọn tham số thích hợp cho biểu thức **copy(\_\_\_,\_\_\_,\_\_\_)** để khi thực hiện kết quả sẽ là **’Ngoc’**:
157. hoten, 12, 4
158. hoten, 4, 12
159. 12, 4, hoten
160. 4, 12, hoten
161. Hàm **upcase(ch)** có công dụng:
162. Cho chữ cái in hoa.
163. Cho chữ cái in thường.
164. Cho chữ cái in hoa ứng với chữ cái trong *ch*.
165. Cho chữ cái in thường ứng với chữ cái trong *ch*.
166. Hàm **pos(s1, s2)** có công dụng:
167. Cho vị trí xuất hiện của xâu *s1* trong xâu *s2*.
168. Cho vị trí xuất hiện của xâu *s2* trong xâu *s1*.
169. Cho vị trí xuất hiện đầu tiên của xâu *s1* trong xâu *s2*.
170. Cho vị trí xuất hiện đầu tiên của xâu *s2* trong xâu *s1*.
171. Cho **s1 := ‘EF’;** **s2 := ‘abcdEfgEF’;** khi đó biểu thức **pos(s1, s2)** sẽ có giá trị là:
172. 5
173. 8
174. 56
175. 89
176. Cho **s1 := ‘abH’;** **s2 := ‘abcdEFgEF’;** khi đó biểu thức **pos(s1, s2)** sẽ có giá trị là:
177. 0
178. 1
179. 12
180. Báo lỗi.
181. Cho trước **ch := ‘d’;** khi đó kết quả của biểu thức **upcase(ch)** là:
182. ‘D’
183. “D”
184. ‘d’
185. “d”
186. Biểu thức nào sau đây sẽ cho kết quả là **’E’**:
187. *upcase(’e’)*
188. *upcase(’E’)*
189. *upcase(ch)*, biết biến *ch* đang lưu trữ giá trị ’e’.
190. Tất cả đáp án A, B và C đều đúng.
191. Sau khi thực hiện đoạn lệnh:



thì trên màn hình hiển thị nội dung:

1. Ket qua: Ti
2. Ket qua: Teo
3. Ket qua: iT
4. Ket qua: oeT
5. Sau khi thực hiện đoạn lệnh:



thì trên màn hình hiển thị nội dung:

1. 11A4
2. 11 A 4
3. Ket qua: 11A4
4. Ket qua: 11 A 4

**Phần 2. Trả lời**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Câu**  **Đáp án** | **01** | **02** | **03** | **04** | **05** | **06** | **07** | **08** | **09** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** |
| **A** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **B** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **C** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **D** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **Câu**  **Đáp án** | **21** | **22** | **23** | **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** |
| **A** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **B** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **C** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |
| **D** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** | **** |