| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | What will be the output of the program ?<br>**#include<stdio.h>**<br><br>int main()<br>{<br>int a[5] = {5, 1, 15, 20, 25};<br>int i, j, m;<br>i = ++a[1];<br>j = a[1]++;<br>m = a[i++];<br>printf("%d, %d, %d", i, j, m);<br>return 0;<br>} |
| ((OPTION_A)) | 2,1,15 |
| ((OPTION_B)) | 1,2,5 |
| ((OPTION_C)) | 3,2,15 |
| ((OPTION_D)) | 2,3,20 |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | **Step 1**: int a[5] = {5, 1, 15, 20, 25}; The variable arr is declared as an integer array with a size of 5 and it is initialized to<br>a[0] = 5, a[1] = 1, a[2] = 15, a[3] = 20, a[4] = 25 .<br>**Step 2**: int i, j, m; The variable i,j,m are declared as an integer type.<br>**Step 3**: i = ++a[1]; becomes i = ++1; Hence i = 2 and a[1] = 2<br>**Step 4**: j = a[1]++; becomes j = 2++; Hence j = 2 and a[1] = 3.<br>**Step 5**: m = a[i++]; becomes m = a[2]; Hence m = 15 and i is incremented by 1(i++ means 2++ so i=3)<br>**Step 6**: printf("%d, %d, %d", i, j, m); It prints the value of the variables i, j, m<br>Hence the output of the program is 3, 2, 15 |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | In C, if you pass an array as an argument to a function, what actually gets passed? |
| ((OPTION_A)) | Value of elements in array |

| | |
|---|---|
| ((OPTION_B)) | First element of the array |
| ((OPTION_C)) | Base address of the array |
| ((OPTION_D)) | Address of the last element of array |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | The statement 'C' is correct. When we pass an array as a funtion argument, the base address of the array will be passed. |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following statements are correct about 6 used in the program?<br>int num[6];<br>num[6]=21; |
| ((OPTION_A)) | In the first statement 6 specifies a particular element, whereas in the second statement it specifies a type. |
| ((OPTION_B)) | In the first statement 6 specifies a array size, whereas in the second statement it specifies a particular element of array. |
| ((OPTION_C)) | In the first statement 6 specifies a particular element, whereas in the second statement it specifies a array size. |
| ((OPTION_D)) | In both the statement 6 specifies array size. |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | The statement 'B' is correct, because int num[6]; specifies the size of array and num[6]=21; designates the particular element($7^{th}$element) of the array. |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | What does the following declaration mean?<br>int (*ptr)[10]; |
| ((OPTION_A)) | ptr is array of pointers to 10 integers |
| ((OPTION_B)) | ptr is a pointer to an array of 10 integers |
| ((OPTION_C)) | ptr is an array of 10 integers |

| | |
|---|---|
| ((OPTION_D)) | ptr is an pointer to array |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following statements are correct about an array?<br>1. The array int num[26]; can store 26 elements.<br>2. The expression num[1] designates the very first element in the array.<br>3. It is necessary to initialize the array at the time of declaration.<br>4. The declaration num[SIZE] is allowed if SIZE is a macro. |
| ((OPTION_A)) | 1 |
| ((OPTION_B)) | 1,4 |
| ((OPTION_C)) | 2,3 |
| ((OPTION_D)) | 2,4 |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | 1. The array int num[26]; can store 26 elements. This statement is true.<br>2. The expression num[1] designates the very first element in the array. This statement is false, because it designates the second element of the array.<br>3. It is necessary to initialize the array at the time of declaration. This statement is false.<br>4. The declaration num[SIZE] is allowed if SIZE is a macro. This statement is true, because the MACRO just replaces the symbol SIZE with given value.<br>Hence the statements '1' and '4' are correct statements. |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | If the two strings are identical, then strcmp() function returns |
| ((OPTION_A)) | -1 |
| ((OPTION_B)) | 1 |
| ((OPTION_C)) | 0 |
| ((OPTION_D)) | Yes |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | strcmp(const char *s1, const char*s2);<br>The strcmp return an int value that is<br>if s1 < s2 returns a value < 0<br>if s1 == s2 returns 0<br>if s1 > s2 returns a value > 0 |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following function is used to find the first occurrence of a given string in another string? |
| ((OPTION_A)) | strchr() |
| ((OPTION_B)) | strrchr() |
| ((OPTION_C)) | strstr() |
| ((OPTION_D)) | strnset() |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | char *strstr(const char *s1, const char *s2);<br>**Return Value**:<br>On success, strstr returns a pointer to the element<br>in s1 where s2 begins (points to s2 in s1).<br>On error (if s2 does not occur in s1), strstr returns null. |

| | |
|---|---|
| | **Example**:<br>**#include <stdio.h>**<br>**#include <string.h>**<br>int main(void)<br>{<br>char *str1 = "IndiaBIX", *str2 = "ia", *ptr;<br>ptr = strstr(str1, str2);<br>printf("The substring is: %s\n", ptr);<br>return 0;<br>}<br>**Output**: The substring is: iaBIX |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | The library function used to find the last occurrence of a character in a string is |
| ((OPTION_A)) | strnstr() |
| ((OPTION_B)) | laststr() |
| ((OPTION_C)) | strrchr() |
| ((OPTION_D)) | strstr() |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | **Declaration**: char *strrchr(const char *s, int c);<br>It scans a string s in the reverse direction, looking for a specific character c.<br>**Example**:<br>**#include <string.h>**<br>**#include <stdio.h>**<br><br>int main(void)<br>{<br>char text[] = "I learn through IndiaBIX.com";<br>char *ptr, c = 'i'; |

| | |
|---|---|
| | ptr = strrchr(text, c);<br>if (ptr)<br>printf("The position of '%c' is: %d\n", c, ptr-text);<br>else<br>printf("The character was not found\n");<br>return 0;<br>}<br>**Output**:<br>The position of 'i' is: 19 |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | How will you print \n on the screen? |
| ((OPTION_A)) | printf("\n"); |
| ((OPTION_B)) | echo "\\n"; |
| ((OPTION_C)) | printf('\n'); |
| ((OPTION_D)) | printf("\\n"); |
| ((CORRECT_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) | The statement printf("\\n"); prints '\n' on the screen. |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following data structures cannot store non-homogeneous elements? |
| ((OPTION_A)) | Arrays |
| ((OPTION_B)) | Structure |
| ((OPTION_C)) | Linked List |

| | |
|---|---|
| ((OPTION_D)) | File |
| ((CORRECT_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | In arrays ____ and _____ are costly but ____ is easy operation |
| ((OPTION_A)) | Searching, insertion, deletion |
| ((OPTION_B)) | Insertion, deletion, searching |
| ((OPTION_C)) | Deletion, searching, insertion |
| ((OPTION_D)) | None of these |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | void main()<br>{<br>int a[5]={1,2};<br>printf("\n%d%d%d",a[2],a[3],a[4]);<br>}<br><br>What will be the output? |
| ((OPTION_A)) | 1 2 2 |

| | |
|---|---|
| ((OPTION_B)) | 2 1 1 |
| ((OPTION_C)) | 0 0 0 |
| ((OPTION_D)) | Garbage Value |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Find the output:<br>void main()<br>{<br>int i=0,a[3];<br>a[i]=i++;<br>printf("%d",a[i]);<br>} |
| ((OPTION_A)) | 0 |
| ((OPTION_B)) | 1 |
| ((OPTION_C)) | Garbage value |
| ((OPTION_D)) | Syntax error |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | While passing the array as actual argument, the function call must have array name_____ |
| ((OPTION_A)) | alone |

| | |
|---|---|
| ((OPTION_B)) | With empty braces |
| ((OPTION_C)) | With its size |
| ((OPTION_D)) | None of these |
| ((CORRECT_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which code will run faster?<br>1. for(i=0;i<100;i++)<br>for(j=0;j<10;j++)<br>a[i][j]=0;<br>2. for(i=0;i<10;i++)<br>for(j=0;j<100;j++)<br>a[i][j]=0; |
| ((OPTION_A)) | Code 1 |
| ((OPTION_B)) | Code 2 |
| ((OPTION_C)) | Both run equally |
| ((OPTION_D)) | None of these |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Consider an integer array int arr[4][5]. If base address is 1020, find the resentation. Size of int is 2 bytes. |
| ((OPTION_A)) | 1020 |
| ((OPTION_B)) | 1038 |
| ((OPTION_C)) | 1039 |

| | |
|---|---|
| ((OPTION_D)) | 1058 |
| ((CORRECT_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Consider the statement int Val[2][4]={1,2,3,4,5,6,7,8}. The element 4 will be at____ |
| ((OPTION_A)) | Val[0][3] |
| ((OPTION_B)) | Val[0][4] |
| ((OPTION_C)) | Val[1][1] |
| ((OPTION_D)) | None of these |
| ((CORRECT_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | void main()<br>{<br>int a[2][3]={2,3};<br>printf("\n%d%d%d%d",a[0][0],a[0][1],a[1][0],a[1][1]);<br>}<br><br>What will be the output? |
| ((OPTION_A)) | 0 0 2 3 |
| ((OPTION_B)) | 2 3 0 0 |
| ((OPTION_C)) | 2 0 3 0 |
| ((OPTION_D)) | 2 0 0 3 |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |

| | |
|---|---|
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | The getchar() library function returns_____ |
| ((OPTION_A)) | Character when any key is pressed |
| ((OPTION_B)) | Character when enter key is pressed |
| ((OPTION_C)) | Display character on the screen when any key is pressed |
| ((OPTION_D)) | None of these |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Find the output:<br>void main()<br>{<br>printf(“%c”,100);<br>} |
| ((OPTION_A)) | Prints 100 |
| ((OPTION_B)) | Prints ASCII equivalent of 100 |
| ((OPTION_C)) | Prints garbage value |
| ((OPTION_D)) | Syntax error |
| ((CORRECT_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the foolowing is more appropriate for reading a multi-word string? |
| ((OPTION_A)) | printf |
| ((OPTION_B)) | scanf |
| ((OPTION_C)) | put |
| ((OPTION_D)) | gets |
| ((CORRECT_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Find the output:<br>int main()<br>{<br>char p[]=”%c\n”;<br>p[1]=’d’;<br>printf(p,65);<br>return 0;<br>} |
| ((OPTION_A)) | a |
| ((OPTION_B)) | A |
| ((OPTION_C)) | c |
| ((OPTION_D)) | 65 |
| ((CORRECT_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |

| | |
|---|---|
| ((QUESTION)) | Sparse matrix have_____ |
| ((OPTION_A)) | Many zero entries |
| ((OPTION_B)) | Many non zero entries |
| ((OPTION_C)) | High dimension |
| ((OPTION_D)) | Many negative entries |
| ((CORRECT_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | The sequential representation of sparse matrix is given by |
| ((OPTION_A)) | stack |
| ((OPTION_B)) | Queues |
| ((OPTION_C)) | Arrays |
| ((OPTION_D)) | Linked List |
| ((CORRECT_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) | |

| | |
|---|---|
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | If polynomial is 5x^3+3x^2+10x+2, the degree is |
| ((OPTION_A)) | 3 |
| ((OPTION_B)) | 2 |
| ((OPTION_C)) | 1 |
| ((OPTION_D)) | 0 |
| ((CORRECT_CHOICE)) (A/B/C/D) | A |

| | |
|---|---|
| ((EXPLANATION))<br>(OPTIONAL) | |