|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <iostream>  using namespace std;  int main()  {  int a[2][4] = {3, 6, 9, 12, 15, 18, 21, 24};  cout << \*(a[1] + 2) << \*(\*(a + 1) + 2) << 2[1[a]];  return 0;  } |
| ((OPTION\_A)) | 151821 |
| ((OPTION\_B)) | 212121 |
| ((OPTION\_C)) | 242424 |
| ((OPTION\_D)) | Compile time error |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | void main()  {  int m, i = 0, j = 1, k = 2;  m = i++ || j++ || k++;  printf("%d %d %d %d", m, i, j, k);  } |
| ((OPTION\_A)) | 1 1 2 3 |
| ((OPTION\_B)) | 1 1 2 2 |
| ((OPTION\_C)) | 0 1 2 2 |
| ((OPTION\_D)) | 0 1 2 3 |
| ((CORRECT\_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | What is the return type of malloc() or calloc() |
| ((OPTION\_A)) | void \* |
| ((OPTION\_B)) | int \* |
| ((OPTION\_C)) | void \*\* |
| ((OPTION\_D)) | Pointer of allocated memory location |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following statement is correct? |
| ((OPTION\_A)) | The default value for an argument cannot be function call |
| ((OPTION\_B)) | C++ allows the redefinition of a default parameter |
| ((OPTION\_C)) | Both A and B |
| ((OPTION\_D)) | C++ does not allow the redefinition of a default parameter |
| ((CORRECT\_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which of the following can be overloaded? |
| ((OPTION\_A)) | Objects |
| ((OPTION\_B)) | Functions |
| ((OPTION\_C)) | Operators |
| ((OPTION\_D)) | Both B and C |
| ((CORRECT\_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Specify the 2 library functions to dynamically allocate memory? |
| ((OPTION\_A)) | malloc() and mealloc() |
| ((OPTION\_B)) | alloc() and mealloc() |
| ((OPTION\_C)) | malloc() and calloc() |
| ((OPTION\_D)) | memalloc() and faralloc() |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Point out the error in the program  struct emp  {  int ecode;  struct emp e;  }; |
| ((OPTION\_A)) | Error: in structure declaration |
| ((OPTION\_B)) | Linker error |
| ((OPTION\_C)) | No error |
| ((OPTION\_D)) | Run time error |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <iostream>  using namespace std;  int main()  {  for (int i = 10; i > 6; i = i - 2)  cout << i;  for (int i = -5; i > -7; i--)  cout << i + 1;  return 0;  } |
| ((OPTION\_A)) | 10 8 6 -5 -6 |
| ((OPTION\_B)) | 10 8 -4 -5 |
| ((OPTION\_C)) | 10 8 -5 -6 |
| ((OPTION\_D)) | 8 6 -4 -5 |
| ((CORRECT\_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | Which member function is used to determine whether the stream object is currently associated with a file? |
| ((OPTION\_A)) | is\_open |
| ((OPTION\_B)) | buf |
| ((OPTION\_C)) | string |
| ((OPTION\_D)) | none of the above |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 2 |
| ((QUESTION)) | #include <iostream>  using namespace std;  class Test {  static int i;  int j;  }  int Test::i;  int main() {  cout<<sizeof(Test);  return 0;  }  Assume size of int as 4 bytes |
| ((OPTION\_A)) | 4 |
| ((OPTION\_B)) | 8 |
| ((OPTION\_C)) | Compile Error |
| ((OPTION\_D)) | Run time Error |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | When can we have two classes with same name? |
| ((OPTION\_A)) | Not possible |
| ((OPTION\_B)) | In different files |
| ((OPTION\_C)) | Different namespace |
| ((OPTION\_D)) | Anywhere |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <iostream>  using namespace std;  class X {  public: X()  { cout<<"X"; }  ~X()  { cout<<"~X"; }  };  class Y : public X {  public: Y()  { cout<<"Y"; }  ~Y()  { cout<<"~Y"; }  };  int main() {  Y obj;  return 0;  } |
| ((OPTION\_A)) | XY~X~Y |
| ((OPTION\_B)) | XY~Y~X |
| ((OPTION\_C)) | X~XY~Y |
| ((OPTION\_D)) | X~X~YY |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 2 |
| ((QUESTION)) | #include <stdio.h>  int main()  {  int a = 1, b = 2, c = 3;  char d = 0;  if (a, b, c, d) {  printf("enter in the if\n");  }  printf("not enterd\n");  return 0;  } |
| ((OPTION\_A)) | enter in the if |
| ((OPTION\_B)) | not entered |
| ((OPTION\_C)) | run time error |
| ((OPTION\_D)) | segmentation fault |
| ((CORRECT\_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | int main()  {  cout<<"Hello";;;;;  return 0;  } |
| ((OPTION\_A)) | Compilation Error |
| ((OPTION\_B)) | Runtime Error |
| ((OPTION\_C)) | Hello |
| ((OPTION\_D)) | Hello;;;; |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | The two statements that can be used to change the flow of control are \_\_\_\_\_\_\_\_ |
| ((OPTION\_A)) | if and while |
| ((OPTION\_B)) | if and switch |
| ((OPTION\_C)) | switch and do-while |
| ((OPTION\_D)) | break and continue |
| ((CORRECT\_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | If p and q are assigned values 2 and 3 respectively then the statement p=q++ |
| ((OPTION\_A)) | Gives an error message |
| ((OPTION\_B)) | Assigns a value 4 to p |
| ((OPTION\_C)) | Assigns a value 3 to p |
| ((OPTION\_D)) | Assigns a value 5 to p |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 2 |
| ((QUESTION)) | #include <iostream>  using namespace std;  #include <iostream>  using namespace std;  int main()  {  int i, x[5], y, z[5];  for (i = 0; i < 5; i++) {  x[i] = i;  z[i] = i + 3;  y = z[i];  x[i] = y++;  }  for (i = 0; i < 5; i++)  cout << x[i] << " ";  return 0;  } |
| ((OPTION\_A)) | 3 4 5 6 7 |
| ((OPTION\_B)) | 4 5 6 7 8 |
| ((OPTION\_C)) | 2 3 4 5 6 |
| ((OPTION\_D)) | none of above |
| ((CORRECT\_CHOICE)) (A/B/C/D) | B |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <stdio.h>  #define square(x) (x \* x)  int main()  {  int x, y = 1;  x = square(y + 1);  printf("%d\n", x);  return 0;  } |
| ((OPTION\_A)) | Error |
| ((OPTION\_B)) | 4 |
| ((OPTION\_C)) | 3 |
| ((OPTION\_D)) | Garbage Value |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | If a variable is a pointer to a structure, then which of the following operator is used to access data members of the structure through the pointer variable? |
| ((OPTION\_A)) | . |
| ((OPTION\_B)) | \* |
| ((OPTION\_C)) | & |
| ((OPTION\_D)) | -> |
| ((CORRECT\_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <stdio.h>  int main()  {  int i=3, \*j, k;  j = &i;  printf(“%d\n”, i\*\*j\*i+\*j);  return 0;  } |
| ((OPTION\_A)) | 30 |
| ((OPTION\_B)) | 27 |
| ((OPTION\_C)) | 9 |
| ((OPTION\_D)) | 3 |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | For automatic objects, constructors and destructors are called each time the objects |
| ((OPTION\_A)) | Enter and leave scope |
| ((OPTION\_B)) | Inherit parent class |
| ((OPTION\_C)) | Are Constructed |
| ((OPTION\_D)) | Are destroyed |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 2 |
| ((QUESTION)) | #include <iostream>  using namespace std;  int n(char, int);  int (\*p) (char, int) = n;  int main()  {  (\*p)('d', 9);  p(10, 9);  return 0;  }  int n(char c, int i)  {  cout << c << i;  return 0;  } |
| ((OPTION\_A)) | d9  9 |
| ((OPTION\_B)) | d9d9 |
| ((OPTION\_C)) | d9 |
| ((OPTION\_D)) | d9109 |
| ((CORRECT\_CHOICE)) (A/B/C/D) | A |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | void main()  {  int a = 1;  switch(a)  {  case 1: cout<<"One";  case 2: cout<<"Two";  case 3: cout<<"Three";  default: cout<<"Default";  }  } |
| ((OPTION\_A)) | One |
| ((OPTION\_B)) | Compilation Error |
| ((OPTION\_C)) | Default |
| ((OPTION\_D)) | OneTwoThree |
| ((CORRECT\_CHOICE)) (A/B/C/D) | D |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 1 |
| ((QUESTION)) | #include <stdio.h>  int main()  {  int val = 1;  do {  val++;  ++val;  } while (val++ > 25);  printf("%d\n", val);  return 0;  } |
| ((OPTION\_A)) | 25 |
| ((OPTION\_B)) | 3 |
| ((OPTION\_C)) | 4 |
| ((OPTION\_D)) | 26 |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |

|  |  |
| --- | --- |
| ((MARKS)) (1/2/3...) | 2 |
| ((QUESTION)) | #include <stdio.h>  void main()  {  printf(6 + "Geeks for Geeks");  } |
| ((OPTION\_A)) | 6Geeks for Geeks |
| ((OPTION\_B)) | Geeks for Geeks |
| ((OPTION\_C)) | for Geeks |
| ((OPTION\_D)) | Compile time error |
| ((CORRECT\_CHOICE)) (A/B/C/D) | C |
| ((EXPLANATION)) (OPTIONAL) |  |