Question 1

Which of the function declarations below is the fastest way to pass a struct type called*MyStruct* to a function?

1.  void F(struct MyStruct\* s);
2.  void F(MyStruct\* s);
3.  void F(struct MyStruct s);
4.  void F(MyStruct s);
5. You answered this question correctly.

Question 2

Which statement is false about the pre-processor?

1.  The pre-processor can insert another file in a source file.
2.  Pre-processor instructions start with '#' and end with a ';'
3.  The pre-processor adapts your source file before it is processed by the compiler.
4.  The pre-processor can replace an identifier with other text.
5. You answered this question correctly.

Question 3

Which two of the options below declares correctly a two-dimensional array (matrix) of integers?

1.  int m[2][2]={1, 2, 3, 4};
2.  int m[4, 4];
3.  int m[3][3];
4.  int m[][]={{1, 2}, {3, 4}};
5. You answered this question correctly.

Question 4

Which combination of pre-processor statements enable conditional compilation?

1.  #ifdef, #else, #endif
2.  #ifdefined, #else, #endif
3.  #ifnotdef, #else, #endif
4.  #define, #elseif, #endif
5. You answered this question correctly.

Question 5

How do you assign the address of an *int* variable to an *int\** pointer variable?

1.  int i; int\* ip=address\_of(i);
2.  int i; int\* ip=&i;
3.  int i; int\* ip=addressof(i);
4.  int i; int\* ip=\*i;
5. You answered this question correctly.

Question 6

Which statement is true about the following program?

Code (C++):

1. #define sqr(x) (x \* x)
3. int main()
4. {
5. printf("%.2f, ", sqr(10.0));
6. printf("%.2f**\n**", sqr(10.0-5.0));
7. }

1.  This program does not compile because the define is missing the semi-colon (;) at the end of the line.
2.  This program does not compile because the define is missing round brackets around the x.
3.  The output of this program is: 100.00, 25.00
4.  The output of this program is: 100.00, -45.00

Question 7

Which of the following options creates a copy of the string declared below?

1.  char s2[80]; strcpy(s2, s1);
2.  char\* s2=s1;
3.  char s2[]=s1;
4.  char\* s2; strcpy(s2, s1);

Question 8

When having the following C code, what code can be placed instead of the ... to print the member *x* of the struct variable.

Code (C++):

1. struct Point
2. {
3. double x;
4. double y;
5. };
7. int main()
8. {
9. struct Point p, \*pp=&p;
10. printf("%.2f**\n**", ...);
11. }

1.  pp->x
2.  pp.x
3.  The struct variable can't be read because it is not initialised.
4.  pp::x

Question 9

Which statement is false about console input?

1.  The function *scanf()* can convert strings read from the standard input into the correct type. For that it needs format specifiers similar as those for the *printf()* function.
2.  When a single *scanf()* statement does not read all data from a line entered on the console, the rest of the line will be discarded.
3.  The standard input is by default the console (keyboard) but it can be redirected to read from a file.
4.  Console input is buffered and input will only send to the program after the user pressed <enter>. Thus the *getchar()* function reads a single character from the standard input only after the user pressed <enter>.

Question 10

Which of the following statements is the best way to allocate 10 integers on the heap?

1.  int\* arr=(int\*)malloc(10\*int);
2.  int\* arr=(int\*)malloc(10\*sizeof(int));
3.  int\* arr=(int\*)malloc(sizeof(int)[10]);
4.  int\* arr=(int\*)malloc(int[10]);
5. You answered this question correctly.

Question 1

Which statement is true about the following code?

Code (C++):

1. int (\*arr2)[5];

1.  This code declares an array of 5 pointers to an *int*.
2.  This code declares a pointer to an array of *int*s.
3.  This code declares an array of 5 *int*s.
4.  This code does not compile.

Question 2

Which of the option(s) below are correct forms to access the third element of the array in the flowing code?

Code (C++):

1. int array[10]; int i;

1.  i=\*array+2;
2.  i=array[2];
3.  i=(array+2)[0];
4.  i=\*(array+2);
5. You answered this question correctly.

Question 3

Which statement is false about console input?

1.  The function *scanf()* can convert strings read from the standard input into the correct type. For that it needs format specifiers similar as those for the *printf()* function.
2.  Console input is buffered and input will only send to the program after the user pressed <enter>. Thus the *getchar()* function reads a single character from the standard input only after the user pressed <enter>.
3.  The standard input is by default the console (keyboard) but it can be redirected to read from a file.
4.  When a single *scanf()* statement does not read all data from a line entered on the console, the rest of the line will be discarded.
5. You answered this question correctly.

Question 4

Which of the following statements is the best way to allocate 10 integers on the heap?

1.  int\* arr=(int\*)malloc(10\*sizeof(int));
2.  int\* arr=(int\*)malloc(int[10]);
3.  int\* arr=(int\*)malloc(10\*int);
4.  int\* arr=(int\*)malloc(sizeof(int)[10]);
5. You answered this question correctly.

Question 5

When having the following C code, what code can be placed instead of the ... to print the member *x* of the struct variable.

Code (C++):

1. struct Point
2. {
3. double x;
4. double y;
5. };
7. int main()
8. {
9. struct Point p, \*pp=&p;
10. printf("%.2f**\n**", ...);
11. }

1.  pp::x
2.  pp->x
3.  The struct variable can't be read because it is not initialised.
4.  pp.x
5. You answered this question correctly.

Question 6

Which statement is false about the pre-processor?

1.  The pre-processor can insert another file in a source file.
2.  The pre-processor adapts your source file before it is processed by the compiler.
3.  Pre-processor instructions start with '#' and end with a ';'
4.  The pre-processor can replace an identifier with other text.
5. You answered this question correctly.

Question 7

Which statement is true about the following program?

Code (C++):

1. #define sqr(x) (x \* x)
3. int main()
4. {
5. printf("%.2f, ", sqr(10.0));
6. printf("%.2f**\n**", sqr(10.0-5.0));
7. }

1.  This program does not compile because the define is missing the semi-colon (;) at the end of the line.
2.  This program does not compile because the define is missing round brackets around the x.
3.  The output of this program is: 100.00, 25.00
4.  The output of this program is: 100.00, -45.00
5. You answered this question correctly.

Question 8

Which of the following typedef statements below defines a pointer to an array of 10 integers?

1.  typedef int (\*ArrayPointer)[];
2.  typedef int (\*ArrayPointer[]);
3.  typedef int \*ArrayPointer[];
4.  typedef int\*[] ArrayPointer;

Question 9

Which of the *scanf()* statements below read in a string and a integer (separated by a space) in the variables defined as follows?

Code (C++):

1. char str[10];
2. int i;

1.  scanf("%9s %d", str, &i);
2.  You can't use a single *scanf()* statement because the number entered by the user after the string will be read in as part of the string instead of read separately as integer.
3.  scanf("%9s %d", str, i);
4.  scanf("%9s %d", &str, &i);
5. You answered this question correctly.

Question 10

Which two statements are true about typedefs?

1.  Typedefs can be used to make it simpler to port your code to other platforms that have different implementations of certain types.
2.  Typedefs are a compiler construct to give another (simpler) name to a (complex) type definition.
3.  Typedefs can be used to define constant values.
4.  Typedefs are a pre-processor construct to replace an identifier with the specified code.

Question 1

Which two statements are true about typedefs?

1.  Typedefs can be used to define constant values.
2.  Typedefs are a pre-processor construct to replace an identifier with the specified code.
3.  Typedefs are a compiler construct to give another (simpler) name to a (complex) type definition.
4.  Typedefs can be used to make it simpler to port your code to other platforms that have different implementations of certain types.
5. You answered this question correctly.

Question 2

Which statement is false about console input?

1.  Console input is buffered and input will only send to the program after the user pressed <enter>. Thus the *getchar()* function reads a single character from the standard input only after the user pressed <enter>.
2.  The standard input is by default the console (keyboard) but it can be redirected to read from a file.
3.  The function *scanf()* can convert strings read from the standard input into the correct type. For that it needs format specifiers similar as those for the *printf()* function.
4.  When a single *scanf()* statement does not read all data from a line entered on the console, the rest of the line will be discarded.
5. You answered this question correctly.

Question 3

Which combination of pre-processor statements enable conditional compilation?

1.  #ifdefined, #else, #endif
2.  #ifdef, #else, #endif
3.  #define, #elseif, #endif
4.  #ifnotdef, #else, #endif
5. You answered this question correctly.

Question 4

How do you retrieve the length of an array variable defined as: int arr[10]?

1.  Using the *arr.length* data member.
2.  Using the *array\_length(arr)* function.
3.  C does not store the length of an array, so it can't be retrieved.
4.  Using the *sizeof(arr)* operator.

Question 5

Which of the option(s) below are correct forms to access the third element of the array in the flowing code?

Code (C++):

1. int array[10]; int i;

1.  i=\*array+2;
2.  i=array[2];
3.  i=\*(array+2);
4.  i=(array+2)[0];
5. You answered this question correctly.

Question 6

Which of the function declarations below is the fastest way to pass a struct type called*MyStruct* to a function?

1.  void F(MyStruct\* s);
2.  void F(MyStruct s);
3.  void F(struct MyStruct s);
4.  void F(struct MyStruct\* s);
5. You answered this question correctly.

Question 7

Which statement is true about the *malloc()* function?

1.  Memory allocated with *malloc()* must be de-allocated with *free()*.
2.  Memory allocated with *malloc()* must be de-allocated with *delete*.
3.  Memory allocated with *malloc()* must be de-allocated with *release()*.
4.  Memory allocated with *malloc()* is de-allocated automatically.

Question 8

Which statement is true about the following code?

Code (C++):

1. union Name
2. {
3. char firstName[10];
4. char lastName[10];
5. };
7. int main()
8. {
9. union Name name;
10. strcpy(name.firstName, "Dennis");
11. strcpy(name.lastName, "Ritchie");
12. printf("%d", sizeof(name));
13. }

1.  The output of this program is: 8
2.  The output of this program is: 10
3.  The output of this program is: 20
4.  The output of this program is: 11

Question 9

Which statement is true about the following program?

Code (C++):

1. #define sqr(x) (x \* x)
3. int main()
4. {
5. printf("%.2f, ", sqr(10.0));
6. printf("%.2f**\n**", sqr(10.0-5.0));
7. }

1.  The output of this program is: 100.00, -45.00
2.  This program does not compile because the define is missing the semi-colon (;) at the end of the line.
3.  The output of this program is: 100.00, 25.00
4.  This program does not compile because the define is missing round brackets around the x.
5. You answered this question correctly.

Question 10

Which two of the options below declares correctly a two-dimensional array (matrix) of integers?

1.  int m[][]={{1, 2}, {3, 4}};
2.  int m[4, 4];
3.  int m[2][2]={1, 2, 3, 4};
4.  int m[3][3];