1.

What is the output of the following code segment?

```
char *msg;
char *s = "Winter 2017!";

msg = &s[2];
printf("%c, *(msg+3));
```

(A) r **(B)** 2017! **(C)** 0 **(D)** 017! **(E)** None of the above

2. Will the following code compile? If you think that it will not compile then explain why?

```
char *s= "Great view!";
for (i = 0; i < strlen(s); i++) {
   printf("%c",*s);
   s++;
}
```

(A) Yes **(B)** No

3. Will the following code compile? If you think that it will not compile then explain why?

```
char s[25]= "Great view!";
for (i = 0; i < strlen(s); i++) {
   printf("%c",*s);
   s++;
}
```

(A) Yes **(B)** No

3. Given

```
float age[5];
float *data = age;
```

Which of the following statement does not correctly loads data to the array age (there is more than one)?

A.
```
for (i = 0; i < 5; i++) {
   scanf("%f", &age[i]);
  }
```

B.
```
for (i = 0; i < 5; i++) {
   scanf("%f", data);
   data++
  }
```

C.
```
for (data = age; data < data+ 5; data++) {
   scanf("%f", data);
  }
```

D.
```
for (i = 0; i < 5; i++) {
  scanf("%f", age);
}
```

E.
```
for (data = age; data < data+ 5; data++) {
  scanf("%f", &data[0]);
}
```

F.
```
for (i = 0; i < 5; i++) {
  scanf("%f", &age[0]);
}
```

G.
```
for (i = 0; i < 5; i++) {
  scanf("%f", &data[i]);
  data++
}
```

H.
```
for (i = 0; i < 5; i++) {
  scanf("%f", &age[i]);
}
```

4. What is stored in each frame? Circle all the correct answers

a. Local variables
b. Static variables
c. Return address
d. Function parameters
e. Global variables
f. Function code

5. A union is a data type that

A. can only be declared inside a struct type
B. is useful for constructing linked lists
C. can only be used with arrays
D. reserves the same space of memory for its fields
E. None of the above

6. What will be the output of the following code?

```
Short age [6] = {25, 28, 29, 30, 31,32};

int *p = NULL;
p = &age[4];

p += 2;
printf("%d\n", *p);
```

A. 28
B. 29
C. 30
D. 31
E. None of the above

7. Which of the following statements is an array of 10 pointers to an int:

A. int arr[10]*;
B. int *arr[10];
C. int &arr[10];
D. int arr[10]&;
E. int arr[10];
F. None of the above

8. Given the following code, which statement will print True

```
int x = 5;
int y = 6;

int *p = &x;
int *q = &y
int **pp = &p;
int **qq = &q;
```

A. If (*pp == p) printf ("True\n);
B. If (**pp == *q -1) printf ("True\n");
C. If (*q == *p) printf ("True\n");
D. If (*pp == *p) printf ("True\n");
E. If (*pp == x) printf ("True\n");
F. If (**qq == y) printf ("True\n");
G. None of the above

9. What will be the output of the following code?

```
Short age [6] = {25, 28, 29, 30, 31,32};

int *q = NULL;
q = &age[4];
printf("q-p = %d \n", q-p);
```

A. `q-p = 4`
B. `q-p = 5`
C. `q-p = 6`
D. `q-p = 10`
E. None of the above

10. `Which statement creates a NewArray data type that contains 100 doubles`

A. `typedef double[100] NewArray;`
B. `typedef double NewArray;`
C. `typedef double NewArray[100];`
D. `typedef double *NewArray`
E. `None of the above`

11. `A field that is not a pointer in the structure struct X can be`

A. `Only a primitive C data types`
B. `Any valid C datatype including programmer defined datatypes`
C. `Any valid C datatype except the type struct X;`
D. `Any valid C datatype except union;`
E. `None of the above`

12. `Write a function myStrCopy(char *s1, char *s2) that copies string s2 to s1.`

A. `Use array indices to solve the problem.`
B. `Use only pointers to solve the problems`

13. `Which line of code will be printed?`

```
int y = 5;
int *p = &y;
int **x = &p;
```

A. If (x == **p)) printf("It \n");
B. If (*x == *p) printf("is \n");
C. If (y == p) printf("a \n");
D. If (**x == *p) printf("nice day\n");
E. Noting will be printed

14. `Which option below correctly completes the statement - The do-while loop is used when _____`

A. `The programs wants to execute the block of code at least once`

B. the block of code consists of two or more statements

C. a "for" loop cannot be used

D. the program needs to count the number of iterations that block of code is executed

E. None of the above

15. Which option correctly finds the location of an element k in a one-dimensional array of long numbers (array name is arr)

A. Address of element k = arr+ + k

B. Address of element k = arr + k*sizeof(long)

C. Address of element k = arr + k*sizeof(arr)

D. Address of element k = arr + sizeof(long)

E. None of the above

16. The notation *(arr[3]+1) is the same as

A. *arr[3][1]

B. arr[3][1]

C. arr[1][3]

D. arr[3][0]

E. None of the above

17. Let arr be declared as int arr[20] then the statement *(arr+20) = 20;

A. generates a runtime error

B. generates a compiling error (syntax error)

C. sets the last element in the array to 15

D. overwrites the memory beyond the end of the array

E. None of the above

18. What will be the output of the following code? In the system alignment is at multiple of 4, size of int is 4 bytes, size of short is 2 bytes

```
struct student {
   int id;
   short GPA;
};

struct employee {
   short salary;
   long stockOptions;
```

```
}.


union person {
   struct student s;
   struct employee e;
} p1;

p1.e.salary = 55;
p1.s .id = 115;

printf("student id = %d \n",p1.s.id);
printf("person salary = %d \n",p1.e.salary);
```

19. True of False – write beside each option whether it is true or false

1. If the frame stack contains 250 frames then the program has called a recursive function
2. The first function in the frame stack is main()
3. When a program executes then the frame stack must contain at least two functions
4. The scope of a variable is the block of code that it is declared in.
5. Function parameters are stored in the frame stack
6. Static variables that are declared in a function are stored in the frame stack
7. The function malloc() allocates memory from the heap and initializes all the bytes to 0
8. The statement int *p = (int *) malloc(100); allocates memory for 100 integers

20. Given that size of a short is 2 bytes, int 4 bytes, long 4 bytes, float 4 bytes and double is 8 bytes. What is the size of the following union structure if the alignment is a multiple of 4?

```
union x {
   short age;
   long salary;
   double stockOptionValue;
};
```

A. 8
B. 14
C. 16
D. None of the above

21. The statement strcmp(s1, s2) returns 0 if

A. s1 has the same length as s2
B. s1 is the same as s2

C. s1 is NULL or s2 is NULL

D. None of the above

Coding and Code Traverse Questions:

22. Write a function that compares two strings and s1 and s2 and returns the index of the first character that is not equal between the two strings. For example if s1 =”day1” and s2=”key2” then the function needs to return 2.

A. Use array indices to solve the problem.

B. Use only pointers to solve the problems

23. Write a function that two parameters ***source*** and ***destination*** and copies the string ***source*** to a string ***destination***. The function should allocate memory for the destination string. If the operation is successful then the function returns 0, if the function fails it should return 1.

```
Function prototype
int copyString(char **destination, char *source);
```

24. Write a function that takes copies the string ***source*** to a string ***destination***. The function should allocate memory for the destination string. If the operation is successful then the function returns 0, if the function fails it should return 1.

```
Function prototype
int copyString(char **destination, char *source);
```

25. Write a function myStrConcatenate(char *s1, char *s2, char **s3) that copies string concatenate strings s1 and s2 into string s3. For example if s1=”nice “ and s2=”day!” then s3 will be “nice day!”.
The function needs to allocate the exact memory for s3.

26. Write a function that allocates memory for a two-dimensional array arr2D of integers.

```
#define DIM_1 5
#define DIM_2 10
int **arr2D = NULL

// Add code
```

27. What is the output of the following code? In the system alignment is at multiple of 4, size of int is 4 bytes, size of short is 2 bytes

```
struct student {
   int id;
   short GPA;
};

struct employee {
   short id;
   short salary;
}.


union person {
   struct student s;
   struct employee e;
};

printf("sizeof(struct student) =%d \n", sizeof(struct student));
printf("sizeof(union person) =%d \n", sizeof(union person));
```

28. Write a function that accepts a string ***s1*** and a character ***c1*** and returns the index of the first occurrence of ***c1*** in ***s1***. If the character ***c1*** does not exist in the ***s1*** then the function needs to return -1.

29. Which of the two functions correctly compute the length of the input string? Explain what the problem is, if one exists, in each of the two functions

1	**2**

int myStrLength1(char *s)
{
int length =0;
while (*s != '\0') s++;
length = s;
return(length);
}
	int myStrLength2(char *s)
	{
	int length =0;
	for (; s[length]!='\0'; length++,s++);
	return(length);
	}

30. What is the output of the following code segment? In this system the alignment is at a multiple of 4, the size of a short is 2 bytes and the size of an int is 4 bytes.

```
typedef struct car {
       int weight;
       char numDoors;
} tCAR;

typedef struct motorcycle {
       short cost;
       short weight;
} tBIKE;

typedef struct vehicle{
       tCAR car;
```

```
	tBIKE bike;
}tVEHICLE;

typedef union {
	tCAR car;
	tBIKE bike;
} uVEHICLE;


int main()
{
	tVEHICLE v1;
	uVEHICLE v2;

	printf("sizeof(tCAR) = %lu \n", sizeof(tCAR));
	printf("sizeof(tBIKE) = %lu \n", sizeof(tBIKE));
	printf("sizeof(tVEHICLE) = %lu \n", sizeof(tVEHICLE));
	printf("sizeof(uVEHICLE) = %lu \n", sizeof(uVEHICLE));

	v1.car.weight = 500;
	v1.bike.cost = 150;
	printf("v1.car.weight = %d v1.bike.cost = %d\n", v1.car.weight, v1.bike.cost);

	v2.car.weight = 500;
	v2.bike.cost = 150;
	printf("v2.car.weight = %d v2.bike.cost = %d\n", v2.car.weight, v2.bike.cost);
}
```

31. What is the output of the following code segment?

```
typedef struct node{
      struct node *next;
          int  data;
   } Node;

   void foo(struct node* head)
   {

     if(head == NULL)  return;
     printf("%d  ", head->data);

     foo(head->next);
   }
```

What is the output of the function foo(H), where H is the head of the following a linked list H->1->2->3->4->5->NULL

32. Write an iterative code that searches for a given record in a linked list (namely, using a for loop or a while loop). The function accepts as input the head of the list and the number to be searched and returns the last node that contains the number

```
typedef struct node{
      struct node *next;
          int  data;
} Node;
```

For example if the lined list is and the function is called with ***H*** and ***3*** then the function will return the node that is highlighted

H->1->2->3->4->5->3->5->5->NULL

33. Write a recursive code that searches for a given record in a linked list. The function accepts as input the head of the list and the number to be searched and returns the first node that contains the number

```
typedef struct node{
      struct node *next;
          int  data;
} Node;
```

For example if the lined list is and the function is called with ***H*** and ***3*** then the function will return the node that is highlighted

H->1->2->3->4->5->3->5->5->NULL

34. What is a stack? What is a stack-frame?

35. The code below computes that factorial of 3 (namely, 3!). What is the maximum number of stack frames that the stack will have during the execution?

```
1.  int factorial(int n)
2.  {
3.     if (n == 1) return(1);
4.       else return(n* factorial(n-1));
5.  }
6.
7.  int main() {
8.     int rc = factorial(3);
9.  }
```

36. True of false

3.1. A stack must contain at least two functions
3.2. The function main() is the first function in the stack
3.3. The function main() is the second function in the stack
3.4. A program can continue its execution even if the stack is empty
3.5. The make checks if code files need to be compiled again
3.6. If a program is called with no command line parameters then argc=0

37. The parameter argv in the function main(int argc, char *argv[]) is

A. An array of characters
B. A string
C. An array of pointers to character
D. A character pointer
E. None of the above

38. Given the following structure

```
struct node {
      struct node *next;
      int id;
};
```

Write a code to delete the first node from the linked list. Copy the id from the deleted node to the parameter *id*

```
void delete(struct node **head, int *id)

{









}
```

39. Given the following structure

```
struct node {
      struct node *next;
      int id;
};
```

Write a code to inserts a new node as the first node in the list in the linked list and store the ***inputId*** in the node. The function should return the pointer to the new node. If there is no memory then the function should return NULL.

```
struct node *insert(struct node **head, int inputId)

{









}
```