Duration: 2:00 Hours Group: .

Name: Date: .

Data Structure and Algorithms Final Exam

***Part I: answer the following two questions*:**

Q.1 True or False

1. The compiler translates all the program at once and keep a copy of the translated program in a separate file. (√)*
2. A global variable is declared inside of any function (x)
3. Control always return to the caller when the function terminates. (√)
4. If you don't initialize an array of integers, the elements of that array will be set by zero values. (x)
5. High-Level language is more easier and faster than low-level language (x)
6. The switch statement can deal with integer and character data types. (√)
7. In C Programming language, any function can call any function except ***main,*** it could not be called by any another function. (√)
8. The local variables can be accessed by any function. (x)

Q.2 Select the correct answer(s):

1. **When you run the following piece of code, the output will be:**

```
for (i=10 ; i >=0 ;  i -= 5)
{
    printf ("i = %d \t " ,  10-(i-1));
}
```

   a- i = 1 i = 5 i =10
   b- i =1 i = 6 i = 11
   c- i =1 i = 6
   d- i =1 i =2 i =3

2. **In the array below, how can you access the element which has the value 4:**

```
int  arr[3][3]={    {1,2,3}, {4,5,6}, {7,8,9} };
```

   a- arr[0][0]
   b- arr[0][1]
   c- arr[1][0]
   d- arr[1][1]

3. **"The key of …….. is that you have to determine if you are dealing with the data or you are dealing with the address of data"**
   a- Structure
   b- Pointers
   c- Stack
   d- Binary Search Tree

4. **You have the following piece of code:**

```
int x = 0 , y = 4 ;
 while ( x < 11)
 {
      y --;
      x + = 2 * y ;
 }
```

when the loop has finished the value of x is :

a- 1
b- 12
c- 13
d- 14

5. **An array is a collection of variables of:**
   a- Different data types scattered throughout memory
   b- The same data type scattered throughout memory
   c- The same data type placed next to each other in memory
   d- Different data types placed next to each other in memory
6. **You have the following piece of code:**

```
int i;
 for (i = 0; i < 10; i++);
 {
     printf("\t %d", i);
 }
```

The output on the screen will be :

a. 1 2 3 4 5 6 7 8 9 10
b. 0 1 2 3 4 5 6 7 8 9
c. 10
d. none of the above.

7. **When you run the following piece of code, the output will be:**

```
int x=35;
switch(x)
{
      case 20:
        printf("\n value of X < 20 and equal: %d", x);
        break;
      case 30:
        printf("\n value of X > 30 and equal: %d", x);
        break;
      default:
        printf("\n value of X is: %d", x);
        break;
}
```

a- value of X > 30 and equal: 35
b- value of X > 20 and equal: 35

c- value of X is: 35
d- none of the above.

8. **While loop is more appropriate than a for loop when:**
   a- The terminating condition occurs unexpectedly.
   b- The body of the loop will be executed at least once.
   c- the program will be executed at least once.
   d- The number of times the loop will be executed is known before the loop is executed.
9. **Type casting is to:**
   a- Convert a lower type to higher type
   b- Change the type of the variable
   c- Obtain the correct value of an Expression
   d- Make an explicit type conversion.

Part II: Answer the following two questions:

Q.3

a- Write a structure to use it to store data of a customer in a home delivery system of take-away restaurant. The needed data of a customer is the phone number, address, postal code, and customer Name.

```
struct customer
{
        char pNumber[11];
        char Address[30];
        char pCode[5];
        char Name[20];
};
/* number and code are strings because they may start with 0 and integers can't start with 0
(011-xxx-xxx-xx)*/
```

b- Write the line of code to declare an array of your structure with 37 elements.

```
struct customer cust[37];
```

c- Could you make an array of this structure with size N? Where N is an integer variable entered by the user. If yes, write the line of code to do that.

```
//this can be done only with malloc (u can't change array size during run time)
struct customer *pCust;
int n;
printf("Enter Number of Customers : ");
scanf("%d",&n);
pCust = (struct customer *)malloc(sizeof(struct customer)*n);
```

Part III: Answer *only one* of the following two questions:

Q.4

Write the recursion version of the following function:

```
int power(int x, int n)
{
      int p,i;
      p=1;
      for(i=1;i<=n;i++)
      {
            p=p*x;
      }
      return p;
}
```

```
int power (int x, int n)
{
     int p=1;
     if(n)
     {
         p=x*power(x,n-1);
     }
     return p;
}
```