STRUCTURES

In this material, we assume that all necessary header files are included for every code segment.

Short Answer

1. * What is a primitive data type? Why is *structure* not a primitive data type?
2. * While both arrays and *structures* can store multiple values, differentiate *structure* from array.
3. ** Write a structure declaration for each of the following cases.
   - A structure to hold the following data about an e-learning account
     - Account ID (string object)
     - Privilege member (Boolean)
     - Number of courses taken (integer)
     - Average scores (double)

     Define a variable of the structure declared above. Initialize the structure members with the following data: `{ACZ42137-B12-7, True, 3, 9.5}`.
   - A structure to hold the following data about a customer profile on the Grab service
     - Account ID (string object)
     - Privilege member (Boolean)
     - Awarded points (integer)
     - Total amount of money in MOCA wallet (double)

     Define a variable of the structure declared above. Initialize the structure members with the following data: `{ACZ42137-B12-7, True, 3000, 953.234}`.
4. ** Consider the structure declaration shown aside.

```
struct Point{
      int x;
      int y;
};
```

   Write statements to fulfill the following requirements.
   - Define `center` to be a `Point` structure variable with default values.

     Assign 12 to the `x` member and 7 to the `y` member of the variable `center`.

     Display the content of `center`.
   - Define `center` to be a `Point` structure variable and initialize it in either of the cases.
     - Initialize `center` with a given tuple `{x = 12, y = 7}`
     - Initialize `center` with `x = 12` only
     - Initialize with `y = 17` (of course, the other member is arbitrary)

- Consider the following structure pointer variable, `Point *p = nullptr;`

  Dynamically allocate a `Point` structure variable and point `p` to this variable. Use `p` to assign 10 to the `x` member and 14 to the `y` member of structure pointed by `p`.

  Display the content of the structure by accessing from pointer `p`.

5. ** Consider the structure declaration shown aside, which uses the `Point` structure in Question 4.

Write functions to fulfill the following requirements.

```
struct Circle {
     Point center;
     int radius;
};
```

- A function that accepts a `Circle` structure variable as its argument and displays the values of the members of this `Circle` structure variable.
- A function that uses a `Circle` structure reference variable as its parameter and stores the user's input in the members of this `Circle` structure reference variable.
- A function that stores the user's input in the members of a `Circle` structure variable and then returns the structure.

6. ** Consider the `Circle` structure in Question 7. Define `circleArray` to be an array that has 20 elements, each of which is a `Circle` structure, and then write a statement to display the content of the `radius` member of element 19.

7. ** What does the following code segment define `s` to be?

```
struct node{
     int i;
     float j;
};
struct node *s[10];
```

8. ** Identify all possible (syntax/logic/semantic) errors in the following code segment.

-
  ```
  struct site{
       char name[] = "GeeksQuiz";
       int no_of_pages = 200;
  };
  int main(){
       struct site *ptr;
       cout << ptr.no_of_pages << ptr.name;
       return 0;
  }
  ```

-
  ```
  struct st{
       int x;
       struct st next;
  };
  int main(){
       st temp;
       temp.x = 10;
       temp.next = temp;
  ```

```
        cout << temp.next.x;
        return 0;
    }
```

-
  ```
  struct names {
      string last, first;
  }
  int main () {  // display the customer whose name is “Peter Parker”
      names customer = "Peter", "Parker";
      cout << names.first << “ “ << names.last << endl;
      return 0;
  }
  ```

9. ** What does each of the following code segment output?

-
  ```
  struct sec{
      int a;
      char b;
  };
  int main(){
      struct sec s = {25,50};
      struct sec *ps =(struct sec *)&s;
      cout << ps->a << ps->b;
      return 0;
  }
  ```

-
  ```
  struct Test{
      char str[20];
  };
  int main(){
      struct Test st1, st2;
      strcpy(st1.str, "GeeksQuiz");
      st2 = st1;
      st1.str[0] = 'S';
      cout << st2.str;
      return 0;
  }
  ```

-
  ```
  int main(){
      struct ShoeType{
          string style;
          double price;
      };
      ShoeType shoe1, shoe2;
      shoe1.style = "Adidas";
      shoe1.price = 9.99;
      cout << shoe1.style << " $ "<< shoe1.price;
      shoe2 = shoe1;
      shoe2.price = shoe2.price / 9;
      cout << shoe2.style << " $ "<< shoe2.price;
      return 0;
  }
  ```

-
  ```
  struct Time{
      int hours, minutes, seconds;
  };
  int toSeconds(Time now){
      return 3600*now.hours + 60*now.minutes + now.seconds;
  }
  int main(){
      Time t = {5, 30, 45};
      cout << "Total seconds: " << toSeconds(t) << endl;
      return 0;
  }
  ```

-
  ```
  struct MyBox{
        int length, breadth, height;
  };
  void dimension (MyBox M){
        cout << M.length << "x" << M.breadth << "x" << M.height << endl;
  }
  int main (){
        MyBox B1 = {10, 15, 5}, B2, B3;
        ++B1.height;
        dimension(B1);
        B3 = B1;
        ++B3.length;
        B3.breadth++;
        dimension(B3);
        B2 = B3;
        B2.height += 5;
        B2.length--;
        dimension(B2);
        return 0;
  }
  ```

10.*** Assume that a and s are structure (pointer) variables. Describe what each of these expressions, using the *, ->, and . operators, refers.

Expression	Description
`s->m`	
`*a.p`	
`(*s).m`	
`*s->p`	
`*(*s).p`	

Fill-in-the-Blank

1. * Fill in each of the following blanks with an appropriate terminology.

- A structure is a ____________________ data type.
- The _______________ operator is used to access members of a structure variable.
- The _______________ operator is used to access members of a structure pointer variable.
- A _________________ is required after the closing brace of the structure declaration.
- The ______________ symbol will be used when terminating a structure.

2. * Rewrite the following statements using the structure pointer operator.

- `(*rptr).windSpeed = 50;` ____
- `*(*strPtr).num = 10;` ____

3. ** Consider the following statement, `Point p1 = {1}, p2 = {1};`

What does the conditional statement, `(p1 == p2)` yields? _____________________

4. ** Consider the following structure and variable declarations.

```
struct Employee{
    short id;
    int age;
    double wage;
};
struct Company{
    Employee CEO;
    int numberOfEmployees;
};
Company myCompany1, *myCompany2;
```

To access the salary information of the two companies' CEOs,

for `myCompany1`, the statement is ____

for `myCompany2`, the statement is ____

5. ** Consider the following code segment. Fill in the blank so that its output will be `2 1 0`.

```
struct Point{
     int x, y, z;
};
int main(){
     Point p1 = {___________________, ________________, _________________};
     cout << p1.z << " " << p1.y << " " << p1.x;
     return 0;
}
```

6. ** The output of the code segment aside will be

____

```
Point p1[] = { 1, 2, 3, 4 };
cout << p1[1].x;
```

7. ** Consider the structure declaration shown aside.

```
struct CityInfo{
    string cityName,
    string state;
    long population;
    int distance;
};
```

Use a single line to define `location` to be a `CityInfo` structure variable in either of the cases.

- Initialize the `cityName` member with "Asheville", the `state` member with "NC", the `population` with 50000, and the `distance` with 28.

  ____

- Initialize the `cityName` member with "Tampa" only.

  ____

- Initialize `state` with "Arkansas", leaving `population` and `distance` uninitialized.

  ____

- Initialize `cityName` with "Knoxville", `state` with "TN", and `distance` with 90.

  ____

8. ** Consider the following incomplete C++ program.

```
const int N = 100;
struct Thing {
    Thing* x;
};
Thing* allocate_things() {
    Thing* t = new Thing[N];
    for (int i=0; i<N; i++) {
        t[i].x = new Thing;
        t[i].x->x = NULL;
    }
    return t;
}
void deallocate_things(Thing* addr){
    ______________________________________________
    ______________________________________________
    ______________________________________________
}
int main() {
    Thing* addr = allocate_things();
    deallocate_things(addr);
}
```

Complete the function `deallocate_things` so that the program has no memory leak.

9. ** Assume that `cptr` is a pointer to a `Circle` structure. For each the following expressions, identify whether it is equivalent to `cptr->radius`. Briefly explain.

Statements	Equivalence
`*cptr.radius`	______________________
`(*cptr).radius`	______________________
`cptr.(*radius)`	______________________

Consider the following structure declaration.

```
struct addr {
      char city[10];
      char street[30];
      int pin ;
};
```

```
struct {
      char name[30];
      int gender;
      struct addr locate;
} person, *kd = &person;
```

For each the following expressions, identify whether it is equivalent to `*(kd -> name + 2)`. Briefly explain.

Statements	Equivalence
`person.name + 2`	______________________
`kd -> (name + 2)`	______________________
`*((*kd).name + 2)`	______________________

10.*** What is the result of `sizeof(student);` for each of the following code fragments? If the results are not the same, give the reasons.

`struct student{`
`    char a;`
`    char b;`
`    int c;`
`};`
	`struct student{`
	`    char a;`
	`    int c;`
	`    char b;`
	`};`
		`struct student{`
		`    int c;`
		`    char a;`
		`    char b;`
		`};`

True or False

Choose T (True) or F (False) for each of the following statements and then briefly explain in one or two sentences.

1. T F It is optional to put a semicolon after the closing brace of the structure declaration.

2. T F The dot operator is used to access members of a structure variable.

3. T F The structure pointer operator is used to access members of a structure pointer.

4. T F If a structure member is uninitialized, all the members that follow it are also uninitialized.

5. T F It is mandatory to provide initializers for all the members of a structure variable.

6. T F Comparisons on structure variables must be based on individual members.

7. T F Assignments on structure variables must be based on individual members.

8. T F The dot operator has lower precedence than the indirection operator.

9. T F A structure is useful when the program needs to store different values of different types in a single collection.

10. T F A structure can be declared inside a function.

11. T F A structure declaration causes a structure variable to be created.

12. T F Members of a structure variable can be initialized on one line of code.

13. T F Functions do not accept structures as parameters, each member of a structure must be passed individually instead.

14. T F The contents of a structure variable can be displayed by passing the structure variable to the cout object.

15. T F In the following program snippet, both `s1` and `s2` would be variables of structure type defined as below and there won't be any compilation issue.

```
typedef struct Student{
    int rollno;
    int total;
} Student;
```

```
Student s1;
struct Student s2;
```

Algorithm Workbench

1. * Consider the following structure declaration.

```
struct Car {
     string carMake
     string carModel;
     int yearModel;
     double cost;
};
```

Write a definition statement that defines a `Car` structure variable initialized with the given data:

`[carMake: Ford, carModel Model: Mustang, yearModel: 1968, cost: $20,000]`

Define an array that contains 5 elements of `Car` structure type. Initialize the first three elements with the following data

Make	Model	Year	Cost
Ford	Taurus	1997	$ 21,000
Honda	Accord	1992	$ 11,000
Lamborghini	Countach	1997	$200,000

Write a loop that steps through the array and displays the contents of each element.

2. * Declare a structure named `TempScale` with the following members: `fahrenheit` (a double) and `centigrade` (a double).

Next, declare a structure named `Reading`, with the following members: `windSpeed` (an int), `humidity` (a double), and `temperature` (a `TempScale` structure variable)

Next define a `Reading` structure variable, and then

- Write statement(s) that store the following data in the variable you defined, `[windSpeed: 37 mph, Humidity: 32%, fahrenheit: 32 degrees, centigrade: 0 degrees]`
- Write the function `showReading` to accept a `Reading` structure variable as its argument and display the content of the variable on the screen.
- Write the function `referReading` that uses a `Reading` structure reference variable as its parameter and asks the user to enter a value for each structure member.
- Write the function `getReading` that returns a `Reading` structure. The function asks the user to enter a value for each structure member and then returns the structure.
- Write the function `recordReading` that uses a `Reading` structure pointer variable as its parameter. The function asks the user to enter a value for each member of the structure pointed to by the parameter.

3. ** Declare a structure to represent the real part and imaginary part of a complex number. Write code to receive two complex numbers and calculate their sum.

Test Data:

Input the first complex number (real part, imaginary part): 3 2

Input the second complex number (real part, imaginary part): 1 7

Expected Output: 4 + 9i.

Repeat the question for the multiplication of these two complex numbers.

Test Data:

Input the first complex number (real part, imaginary part): 3 2

Input the second complex number (real part, imaginary part): 1 7

Expected Output: -11 + 23i.

4. ** Declare a structure to represent the numerator and denominator of a fraction. Write code to receive two fractions and calculate the subtraction of these two fractions. Note that the resulting fraction must be minimal.

Test Data:

Input the first fraction (numerator, denominator): 5 6

Input the second fraction (numerator, denominator): 1 2

Expected Output: 1 / 3.

Repeat the question for subtraction, multiplication and division (avoid divided by zero).

5. ** Declare a structure to store the names, salary, and hours of work per day of 10 employees. Write code to increase the salary depending on the number of hours of work per day as follows and then print the name of all the employees along with their final salaries.

Hours of work per day	8	10	>=12
Increase in salary	$50	$100	$150

6. ** Declare a structure to represent the two-dimensional coordinate of a point, and then declare another structure to represent three points of a triangle. Write code to receive a triangle and compute its perimeter and area.

Reference: https://www.mathsisfun.com/geometry/herons-formula.html

Test Data:

Input the first point: 0 0

Input the second point: 0 3

Input the third point: 4 0

Expected Output: The perimeter is 12. The area is 6.

7. ** Declare a structure to represent a data, including day, month, and year. Store the current date in the structure. Write code to add 45 days to the current date and display the resulting date.

Test Data: 5 12 2020 (day, month, year)

Expected Output: 19/1/2021

8. *** Declare a structure to store the suit (in ascending order, 0: spade, 1: club, 2: diamond, 3: heart) and rank (in ascending order, 0: Joker, 1: Ace, 2 – 10, 11: Jack, 12: Queen, 13: King) of a card. Write code to receive a set of five cards and determine which hand-ranking the set falls into.

Reference: https://en.wikipedia.org/wiki/List_of_poker_hands

Test Data:

Input the first card (suit, rank): 0 11

Input the second card (suit, rank): 0 10

Input the third card (suit, rank): 0 9

Input the fourth card (suit, rank): 0 8

Input the fifth card (suit, rank): 0 7

Expected Output: Straight flush.

9. *** Let us work on the menu of a library. Create a structure containing book information like accession number, name of author, book title and flag to know whether book is issued or not. Create a menu in which the following can be done.

1 - Display book information

2 - Add a new book

3 - Display all the books in the library of a particular author

4 - Display the number of books of a particular title

5 - Display the total number of books in the library

6 - Issue a book

If we issue a book, then its number gets decreased by 1 and if we add a book, its number gets increased by 1.

10. *** Declare a structure to represent a univariate polynomial, in which each term is represented by its coefficient and exponent. Write code to receive two polynomials and calculate the addition and subtraction of these two polynomials.

Refer to the following link for how to perform arithmetic operations on polynomials.

https://www.mathsisfun.com/algebra/polynomials-adding-subtracting.html