SECJ 1013 PROGRAMMING TECHNIQUE 1
ASSIGNMENT 3

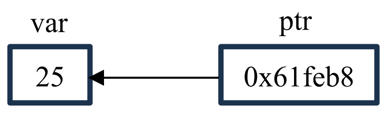
GROUP MEMBER'S (MATRIC NO):
MAXIVIANNA BINTI ROBERT A24CS0109
DAMIA ZAFIRA BINTI NAWAWI A24CS0241

SECTION: 02

1) State whether the following declarations are valid or invalid. Give reasons for the invalid declarations and draw memory layout for the valid declarations. **(7 marks)**

i.
```
int var = 25;
int *ptr = &var;
```
Valid.

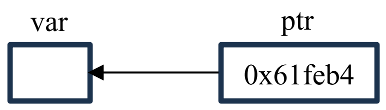
Memory layout:



Memory address of var : 0x61feb8

ii.
```
int var = 30;
int* ptr = var;
```
Invalid. The memory address of var should be assigned to pointer variable ptr by using &.

iii.
```
int var, *ptr;
ptr = &var;
```
Valid.
Memory layout:

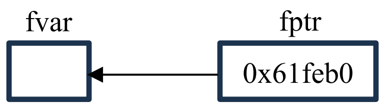


Memory address of var : 0x61feb4

iv.
```
float fvar;
int *ptr = &fvar;
```
Invalid. Data types cannot be mix, the data type of a pointer variable must be the same with the variable it refers to.

v.
```
float fvar, *fptr = &fvar;
```
Valid.

Memory layout:



Memory address of fvar : 0x61feb0

```
vi. int *ptr = &var;
    int var = 25;
```
Invalid. Variable var must be defined before assigned to pointer variable ptr.

```
vii. double* dptr1, dptr2;
     double dvar = 25.2;
     dptr1 = &dvar;
     dptr2 = &dvar;
```
Invalid. The pointer variables should be defined individually,
`double *dptr1, *dptr2;`

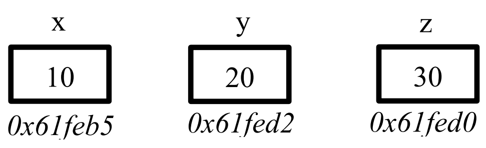
2) Determine the output and draw a memory layout (or memory allocation) of the pointers and variables for code segment below. Note: Draw a memory layout that represents C++ statement line by line. **(7 marks)**

```
int  x = 10, y = 20, z = 30;
int *ptr;

cout << x << " "  << y << " " << z << endl;
ptr = &x;
*ptr *= 10;
ptr = &y;
*ptr *= 4;
ptr = &z;
*ptr *= 2;

cout << x << " "  << y << " " << z << endl;
```

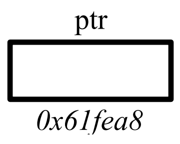
Output:

```
10 20 30
100 80 60
```

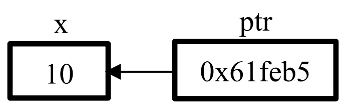
Line 1:



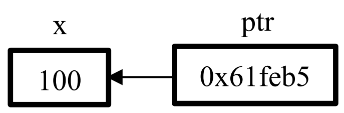
Line 2:



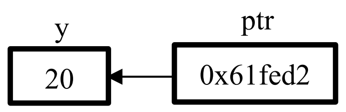
Line 5:



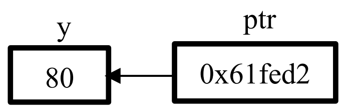
Line 6:



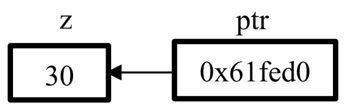
Line 7:



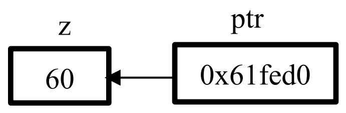
Line 8:



Line 9:



Line 10:



3) Write two statements to free dynamically allocated array and double which are declared as follows: **(2 marks)**

```
int *iPtr = new int [100];
double *dPtr = new double;
      delete [] iPtr;
       delete dPtr;
```

4) Starting address of the following array named iVar is 0xFEC07.

[0]	[1]	[2]	[3]
2	5	8	6

iVar

What is the output that will be displayed based on the following statements? **(4 marks)**

i. cout << iVar;

ii. cout << iVar [0];

iii. cout << *iVar;

iv. cout << *(iVar + 2);

Output:

i.	0xFEC07
ii.	2
iii.	2
iv.	8

5) Write a structure declaration to hold the following data **(6 marks)**

i. About a flight reservation: passenger name, age, reservation code, departure location, destination, flight number, departure time, arrival time, cost and payment status.

```
struct FlightReservation
{
      string name;
      int age;
      string code;
      string departLocation;
      string destination;
      string flightNum;
      char departureTime[6];
      char arrivalTime[6];
      double cost;
      string paymentStatus;
};
```

ii. About saving account: account number, account balance, interest rate, total deposit and total withdraw.

```
struct Saving
{
      char accountNum[30];
      double balance, interest, totalDeposit, totalWithdraw;
};
```

iii. About PT1 assessments: student's name, test 1, assignment, quiz, lab exercise, final exam, course work mark, total mark and grade.

```
struct Assessment
{
      string studName;
      double test1, assignment, quiz, labExec, finalExam, courseWork;
      double totalMark;
      char grade;
};
```

6) A car salesman keeps the information of each model of car he sells. The example of information for 3 cars' models is as in Table 2. Write C++ statement for the following task.

(10 marks)

Model	Engine capacity	Price
Waja	1.6	60000
Wira	1.5	50000
MyVi	1.3	45000

i. Define a structure for storing the above information named Car.

```
struct Car
{
      string model;
      double engineCapacity;
      double price;
};
```

ii. Declare a variable called myCar and initialized it with some values of your choice. Display information on myCar.

```
Car myCar = {"MyVi", 1.3, 45000};

cout << "Car Model : " << myCar.model << endl;

cout << "Engine Capacity : " << myCar.engineCapacity << endl;

cout << "Price : RM" << myCar.price << endl;
```

iii. Declare another variable called mySecondCar and assign values to it using assignment statements. Display information on mySecondCar.

```
Car mySecondCar = {"Waja", 1.6, 60000};

cout << "Car Model : " << mySecondCar.model << endl;

cout << "Engine Capacity : " << mySecondCar.engineCapacity << endl;

cout << "Price : RM" << mySecondCar.price << endl;
```

iv. Print the total of price paid for myCar and mySecondCar.

```
cout << "Total price is RM" << myCar.price + mySecondCar.price << endl;
```

v. Copy the values and information of mySecondCar into myCar and display current information on myCar.

```
myCar.model = mySecondCar.model;

myCar.engineCapacity = mySecondCar.engineCapacity;

myCar.price = mySecondCar.price;

cout << "Car Model : " << myCar.model << endl;

cout << "Engine Capacity : " << myCar.engineCapacity << endl;

cout << "Price : RM" << myCar.price << endl;
```

7) Write the code segment for each of the following tasks: **(8 marks)**

a) Declare a structure type:

i. named `Salary`, with the following members:

`basic` : a double value

`allowances` : a double value

```
struct Salary
{
        double basic;
        double allowances;
};
```

ii. named `Employee`, with the following members:

`name` : a string value
`id` : an integer value

`salary` : a `Salary` structure variable

```
struct Employee
{
        string name;
        int id;
        Salary salary;
};
```

iii.Declare a variable of structure type `Employee` named `myEmp`.

```
Employee myEmp;
```

b) By using the variables and structure declaration in (a), define a function named `displayEmp`. It should accept an `Employee` structure variable as its argument and not return a value. The function should display the contents of the variable onto the

```
Sample output:
Name: Azira
Id: 8902
Basic salary: RM 4500
Allowances: RM 500
```

screen based on figure below. *Notes: Assuming the data for struct members was already assigned.

```
void displayEmp (Employee myEmp)
{
      cout << “Name: ” << myEmp.name << endl;
      cout << “Id: ” << myEmp.id << endl;
      cout << “Basic Salary: ” << myEmp.salary.basic << endl;
      cout << “Allowances: ” << myEmp.salary.allowances << endl;
}
```