Lesson 6 - Exercise 7

Lydia Muñoz

Aybaran Yurtseven

Is all dynamically allocated memory freed?

We began with memory management, verifying that all dynamically allocated memory with new is correctly freed with delete[]. This step is crucial to avoid memory leaks and ensure our program is efficient in its use of resources.

Is all dynamically allocated memory freed?

```
Vehiculo *v=new Vehiculo[tamv];
```

```
delete [] v;
return 0;
```

Is every loop guaranteed to finish?

We analyzed whether all loops guarantee their termination. We observed that each for statement in our code has a clear exit condition, which prevents the program from entering infinite loops. For example, mostrarEnPantalla(), the loop traverses the array to tamv, ensuring that execution does not extend indefinitely.

Is every loop guaranteed to finish?

```
void vehiculos:: mostrarEnPantalla(const Vehiculo *v, int tamv ){
    char c;
    for(int i=0; i<tamv; i++){
        muestraEnPantalla( p: v[i]);
        if((i+1)%5==0){
            cout<<"pulsa enter para los 5siguientes : ";
            cin>>c;
        }

    }
}
```

Can unexpected inputs cause corruption?

We also evaluated whether unexpected input could cause corruption in this function. If a user enters an unexpected value, such as a price less than 0, an exception would be thrown.

Can unexpected inputs cause corruption?

```
cout << "introduce el precio";
cin >> p.precio;
if (p.precio < 0){
    throw string( s: "vehiculos::LeePorTeclado: has introducido mal el precio");
}
```

Is there a possibility of buffer overflow?

Getline(cin, p.marca) is used, which is correct to prevent an overflow.

Is there a possibility of buffer overflow?

```
cout << "introduce la marca";
getline( &: cin >> ws,  &: p.marca);
if (p.marca.length() < 3 || p.marca.length() > 20){
    throw string( s: "vehiculos:: LeePorTeclado: has introducido mal la marca");
}
```

Do all functions and methods calls have the correct number of parameters?

New is used to allocate memory to the array Vehicle *v = new Vehicle[tamv];

It is correctly freed with delete[] v; at the end of main().

Do all functions and methods calls have the correct number of parameters?

```
void vehiculos::LeePorTeclado (Vehiculo &p ){
    cout << "introduce la marca";
    getline( &: cin >> ws, &: p.marca);
    if (p.marca.length() < 3 || p.marca.length() >
```

```
Vehiculo *v=new Vehiculo[tamv];
  posicion= buscarPorMatricula( matr
  cout<<" el vehiculo con matricula
  LeePorTeclado( &: v[posicion]);
catch(const string &e){
```

Are all program variables initialized before use?

All function calls have the
correct number of parameters

Are all program variables initialized before use?

```
int vehiculos:: MaxPrecio(Vehiculo p[], int tamv){
    int aux=0;
    int pos=0;
    for(int i=0; i<tamv; i++){
        if(p[i].precio>aux){
            aux=p[i].precio;
            pos=i;
        }
    }
    return pos;
}
```

THANKS!

Lydia Muñoz

Aybaran Yurtseven