University of Reading

Department of Computer Science

Extending a platform game in C/C++: You can give your extensions a nice title

Firstname Lastname

Module Code	CS1PC20
Assignment report Title	Programming Project
Student Number (e.g. 25098635)	TODO
Date (when the work completed)	23 March 2021
Actual hrs spent for the assignment	TODO
Assignment evaluation (3 key points)	TODO Enter here your comments: engaging task, learned something, could improve the description about XXX

Declaration

I, **Firstname Lastname**, of the Department of Computer Science, University of Reading, confirm that all the sentences, figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that if failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalised accordingly.

Firstname Lastname
February 1, 2021

Contents

1 Introduction

The aim of the project is to assess your ability to analyze, design and implement an extension to an existing C/C++ program. The given program is a platform game that uses the SLD2 library for the graphics.

Your task is to extend the platform game starting from the provided skeleton code. Hence, you must design at least one new feature, then adapt, modify and develop the code to realize this feature. You are free to add any features you wish. You are required to write a report describing the design and implementation of the changes you make. You may develop it under Windows or Linux on the machines in the CS labs or on your own machine.

The assessment will take the form of a report on the game you develop. You will be assessed on the complexity of your contribution and the quality of the report to document this contribution.The code will not be directly assessed although you must include a diff from your gitlab repository from the initial repository to the finished version.

1.1 Learning Objectives

- Designing and implementing larger program in C/C++
- Managing the project development
- Report writing

1.2 Provided Code

You have been provided with a skeleton of a platform game which you imported into the Visual Studio IDE in the tutorial for Week 1. This skeleton uses the SLD2 library for graphics, sound, and keyboard interaction. In Week 5, the tutorial was to follow the online tutorial from `parallelrealities.co.uk` on how the program had been constructed.

The skeleton code is written in C using *structs* as the main data structure. You are free to develop the code in this style or to enrich it using C++ classes using an object orientation style. Either way, you must justify you choice in the report and demonstrate this was a sensible choice for the features you have employed.

1.3 Suggested Approach

You should use the practical sessions A, B and C and the drop-in session on Fridays and the MS Teams channel to access help and support. You may also help and receive help from fellow students. However, you should bare in mind this is an individual project make sure you submit only your own work. See the University guidelines on plagiarism which are online.

Your report should start with an **Introduction** section where you introduce your work. What were the project goals, what features did you wish to include in the game. Think about features of platform games you have played that improve the game play. You may wish to include some of the following, high scores, multiple levels, power-ups, enemies, movement parts *etc*. You should justify what programming style such as C/C++, did you use any OO features *etc*.

2 Design

Document what features you decided to implement and how they interact with the rest of the game. Explain how the game play works. Include a diagram such as a flow chart, Figure 1 or

UML diagram, Figure 2[1].

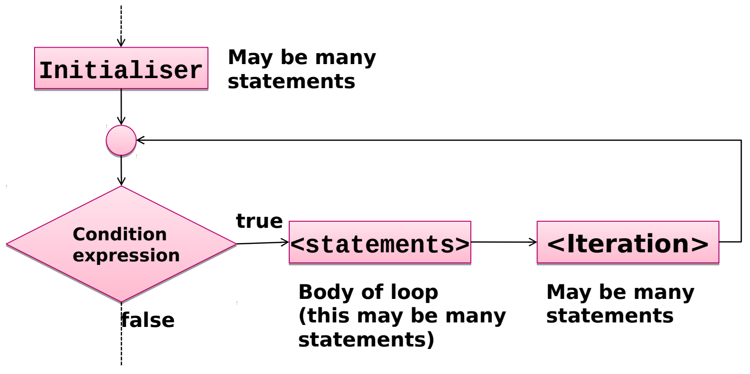

Figure 1: A simple figure

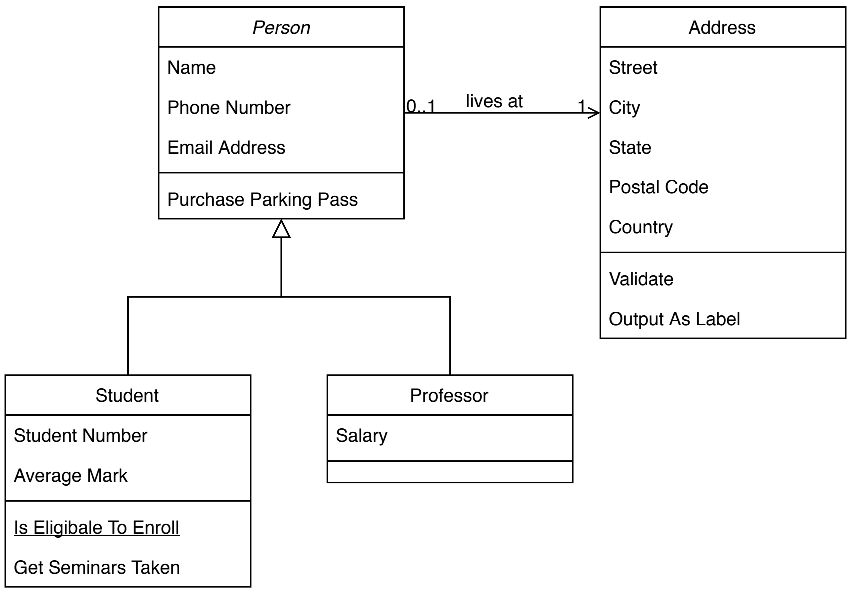

Figure 2: A simple UML class diagram

In Listing 1, you can see how to present a code example. You can refer to specific sections of code using the line numbers, e.g., between Line 5-10, the setup is made, ...

3 Implementation and Development

Describe how you developed the code including what language features you employed and why. How did your program evolve from the original design? Did you adapt the design to improve the game? You should include code fragments and screen shots illustrate particular features.

Describe the finished game, including all the features you added. What features work well? Document any parts of the code which feel are important or enable the program to work in a particular way.

[1]Including both is allowed or even encouraged.

```cpp
static void capFrameRate(long *then, float *remainder){
  long wait, frameTime;
  // This is a comment
  wait = 16 + *remainder;

  *remainder -= (int)*remainder;

  frameTime = SDL_GetTicks() - *then;

  wait -= frameTime;

  if (wait < 1)
  {
    wait = 1;
  }

  SDL_Delay(wait);

  *remainder += 0.667;

  *then = SDL_GetTicks();
}
```

Listing 1: My testcode (example.cpp)

Describe here your journey how you developed the program and reflect about your development. How did you test and debug the program? Did you find all the bugs? What testing have you employed to ensure the game works correctly. Did you use any specific strategies for testing?

4 Conclusion

Summarise your program development. What have your learnt about programming? What have you learnt about developing a significant program. What would either features would you add if you had more time. What would you do differently?

A Code Changes

Repository: `https://csgitlab.reading.ac.uk/di918039/cs1pr-portfolio`.

Reference the URL of your code repository (that you made accessible to us). Include a diff of your code repository on CSGitlab from the provided skeleton code; the intital commit (which is the code for Week 1 tutorial) and your final working version.

This can be achieved as follows: In CSGitlab, go to your repository, select Repository, then Compare. Now enter as Source "master" and as "target" the Git revision of the initial code and press "Compare". You will see the history of commits and the detailed changes (deltas) to the original code. Print this as PDF and include it here.

To make this work: after you received the project code, commit the initial code without any changes.

Source	`master`

...

Target	`81f4d854`

Commits (19)

Examples (/di918039/cs1pr-portfolio/commit/875ad95924237a6ee297bcd7b1dc42bbcf9101be) · 875ad959
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Constructors destructors (/di918039/cs1pr-portfolio/commit/d15b0c59f7cf5e5f962751b82283f811b820686e) · d15b0c59
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Nai (/di918039/cs1pr-portfolio/commit/5d7647ba95ed9c91db55527d0d3ddab19dc75700) · 5d7647ba
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Nai (/di918039/cs1pr-portfolio/commit/5b257911cf5a58be148a343a82465d4174ba7849) · 5b257911
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Further codes (/di918039/cs1pr-portfolio/commit/349cd563fa9ef873da21e3e751de528257464789) · 349cd563
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Further examples (/di918039/cs1pr-portfolio/commit/96ebc50ca5ebee6bf907af03834048d39442276b) · 96ebc50c
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Nai (/di918039/cs1pr-portfolio/commit/486516579e0637c5d05f3aef838a4c868cc8c83a) · 48651657
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Virtual destructors. (/di918039/cs1pr-portfolio/commit/ebf9a368f76f4957498a8e49ed4548e3b1186a9a) · ebf9a368
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Stuff. (/di918039/cs1pr-portfolio/commit/a7f0dc5be115409d6a8c7ff6f92f99c38017ea0d) · a7f0dc5b
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Further (/di918039/cs1pr-portfolio/commit/ecd0b885a3078ff5e14ca46a6e593ba050ddd390) · ecd0b885
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Nai (/di918039/cs1pr-portfolio/commit/6d03b95c0fb17e223c20e593dc99321164d2fbce) · 6d03b95c
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed 3 weeks ago

Nai (/di918039/cs1pr-portfolio/commit/609a7e696101c4daab846d1af87f421f962518b4) · 609a7e69
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed a week ago

Examples for 8 (/di918039/cs1pr-portfolio/commit/f23706f4eaa29c1204e8978c099c7b4975b68364) · f23706f4
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed a week ago

Exception 1 (/di918039/cs1pr-portfolio/commit/3d912c3bc45c05b8b02f64e25c609ba18414d909) · 3d912c3b
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed a week ago

Nai (/di918039/cs1pr-portfolio/commit/320c3ef059540c89c9475673f7a89f5f87c5e3b7) · 320c3ef0
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed a week ago

Smart pointer code (/di918039/cs1pr-portfolio/commit/8e0f875b544d093042bf3176945cccb266e39007) · 8e0f875b
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed about 22 hours ago

Nai (/di918039/cs1pr-portfolio/commit/2a354523c2cdfd16cc6166899422deb1812a76f9) · 2a354523
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed about 21 hours ago

Examples for Lecture 10 (/di918039/cs1pr-portfolio/commit/b174a3397a6fa0ba29df623874cefa4596a5b6d5) · b174a339
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed about 19 hours ago

Examples (/di918039/cs1pr-portfolio/commit/a6d904ef0ea60da9c938dd4ad8d710b40c1476d5) · a6d904ef
Julian M. Kunkel (mailto:juliankunkel@googlemail.com) committed about 17 hours ago

Showing **106 changed files** with **2319 additions** and **0 deletions**

lecture-examples/spring/10/const-expr.cpp 0 → 100644

```
+ #include <iostream>
+
+ using namespace std;
+
+ constexpr int varX = 10;
+
+ constexpr int get_five() {
+   return varX/2;
+ }
+
+ int main(){
+   int some_value[get_five() + 7]; // Create an array of 12 integers
+   return 0;
+ }
```

lecture-examples/spring/10/lambda-simple.cpp 0 → 100644

```
+ #include <string>
+ #include <iostream>
+ #include <algorithm>
+ using namespace std;
+
+ //This example may only work with recent compilers.
+ //clang++-9 -stdlib=libc++ lambda-simple.cpp
+
+ int main() {
+   std::function<string()> funct = []() { return "Hello"; };
+   cout << funct() << endl;
+
+   auto ifunct = []() { return 5; };
+   cout << ifunct() << endl;
+
+
+   int multiplier = 2;
+   auto ifunct2 = [&multiplier](int v) { return v * multiplier; };
+   cout << ifunct2(6) << endl;
+ }
```

lecture-examples/spring/10/lambda.cpp 0 → 100644

```
+ #include <iostream>
+ #include <string>
+ #include <vector>
+
+ using namespace std;
+
+ class Pallet
+ {
+     public:
+         Pallet();
+         Pallet(vector<string> Items) : items(Items) {}
+         int GetWeight() { return items.size(); }
+
+     protected:
+         vector<string> items;
+ };
+
+ int main()
+ {
+     vector<Pallet> pallets = {
+         Pallet({ "Scorpions" }),
+         Pallet({ "Dogs", "Bones", "Biscuits", "Cats", "Food", "Toys" }),
+         Pallet({ "Computers", "Scientists", "Routers", "Monitors" }) };
+
+     sort(pallets.begin(), pallets.end(), [](auto& a, auto& b) {
+       return a.GetWeight() < b.GetWeight();
+     });
+ }
```

lecture-examples/spring/10/macro.cpp 0 → 100644

```
+ #include <string>
+ #include <iostream>
+
+ #define ITEMS \
+ TYPE(Computers), \
+ TYPE(Scientists), \
+ TYPE(Routers), \
+ TYPE(Monitors),
+
+
+ #define TYPE(e) e
+ enum class item {
+ ITEMS
+ };
+ #undef TYPE
+
+ #define TYPE(e) #e
+ const std::string item_strings[] = {
+ ITEMS
+ };
+ #undef TYPE
+
+ int main(){
+   std::cout << item_strings[(int) item::Routers] << std::endl;
+ }
```

lecture-examples/spring/10/meta-gcd.cpp 0 → 100644

```
+ #include <iostream>
+ #include <string>
+ #include <algorithm>
+ using namespace std;
+
+ template< int a, int b > struct GCD {
+     static const int RESULT = GCD< b, a % b >::RESULT;
+ };
+
+ template< int a > struct GCD< a, 0 > {
+     static const int RESULT = a;
+ };
+
+
+
+ int main() {
+   cout << "GCD (25,50) == " << GCD<25, 50>::RESULT << endl;
+   cout << "GCD (12,64) == " << GCD<12, 64>::RESULT << endl;
+   cout << "GCD (a,b) == " << GCD<18, 8398176>::RESULT << endl;
+
+   return 0;
+ }
```