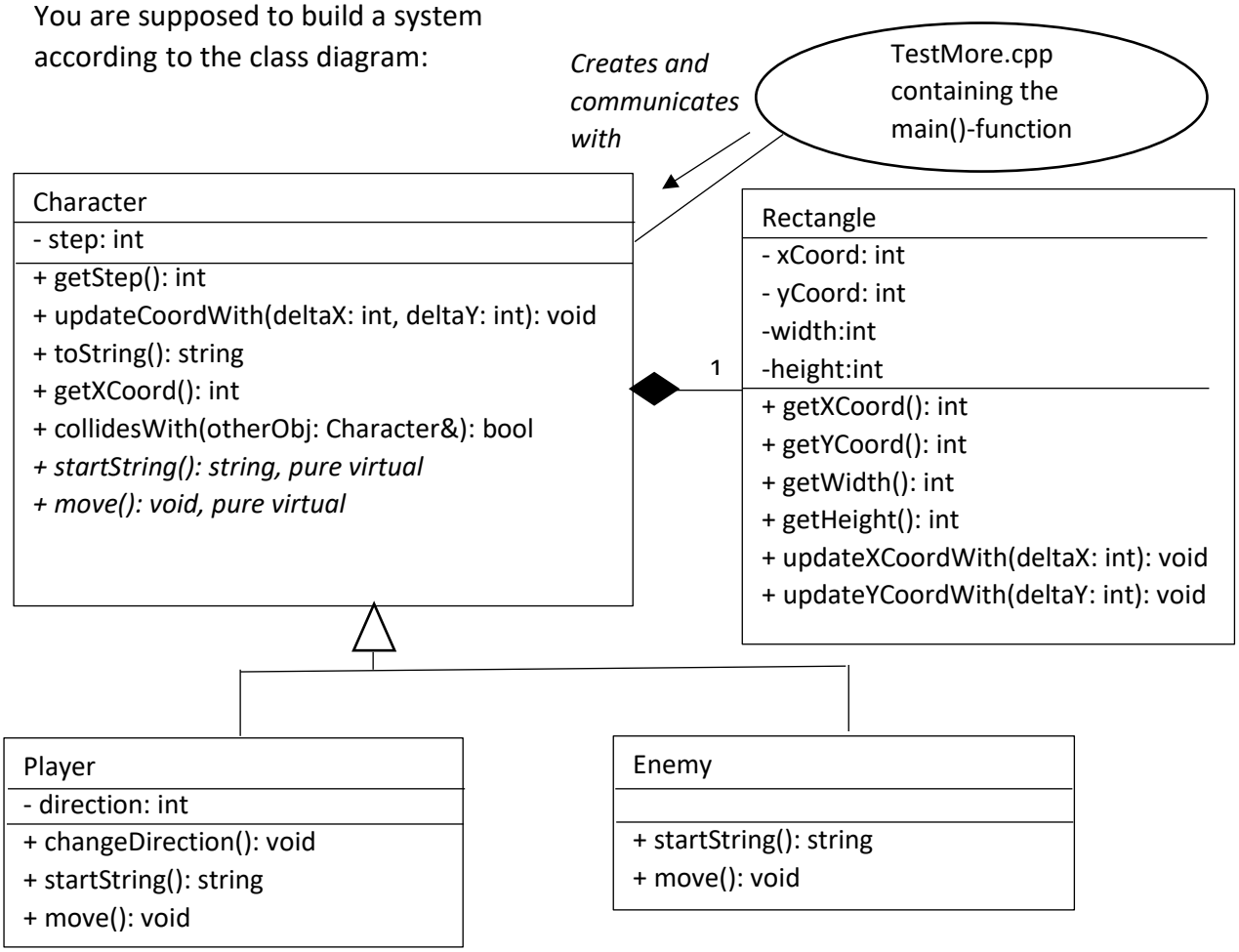
BTH001: Assignment 2

This assignment focuses on inheritance (abstract base class, polymorphism, dynamic binding), type identification and type casting, composition, deep copying and dynamic memory allocation and deallocation.

You are supposed to build a system according to the class diagram:



In this system Player **can only move horisontally** and Enemy **can only move vertically downwards.**

A short description of the classes

Class Rectangle:

xCoord and yCoord give the coordinate of the top left corner

The constructor must have the parameters in the order xCoord, yCoord, width and height

int getWidth(): returns width
int getHeight(): returns height
int getXCoord(): returns the x-coordinate of the top left corner
int getYCoord(): returns the y-coordinate of the top left corner
void updateXCoordWith(int deltaX): *adds* the value of deltaX to the x-coordinate
void updateYCoordWith(int deltaY): *adds* the value of deltaY to the Y-coordinate

Class Character:

step is an integer representing the length of a step (used in each movement)

The constructor must have the parameters in the order xCoord, yCoord, width, height and step

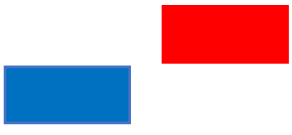
int getStep(): returns the length of a step

void updateCoordWith(int deltaX, int deltaY):
adds, for the rectangle, the value of deltaX to the x-coordinate and the value of deltaY to the y-coordinate

bool collidesWith(const Character& otherObj):
returns true if the rectangle of the calling object overlaps the rectangle of otherObj, otherwise false is returned

Example of overlap:

Example of no overlap:



Description of class Character continues on next page

string toString():
returns a string with a content given by expected outputs from the test programs

int getXCoord(): returns the x-coordinate of the top left corner

The class Player:

Right is the direction when a Player object is created

Movement only in horizontal direction (left, right) with the length of step

For a Player it is not possible to move so far to the left that the x-coordinate gets negative. No restrictions at the right side.

void move(): performs a move

string startString(): returns ”Player”

The class Enemy:

Just movement downwards vertically with the length of step

void move(): performs a move

string startString(): returns ”Enemy”

More info:

You have access to 2 test programs: TestRectangle.cpp and TestCharacter.cpp.
These do not cover all possible situations but give you an indication if you have succeeded with your implementation.
It is a good idea to start with implementing the class Rectangle and then run the test program TestRectangle.cpp.
Continue with the classes in the inheritance hierarchy and then run the test program TestCharacter.cpp.

When your implementation is tested with the test programs mentioned above (and works as expected) do continue with implementing the parts in the file TestMore.cpp which are not completed yet.
Comments in the file TestMore.cpp explain what you are expected to implement.

General requirements:

Private member variables in all classes.
No global variables.
All functions that can be constant shall be constant.
All parameters that ought to be constant shall be constant.
All classes shall be divided into a h-file and a cpp-file with the declaration in the .h-file and the definition in the .cpp-file.

Your solution must not generate any memory leaks.
Use

```
_CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF |_CRTDBG_LEAK_CHECK_DF);
```

in the main()-function to detect memory leaks.