Compte Rendu

Exercice 1

Question 2 :

Les commandes suivantes permettent de de générer les fichier .o, cependant il est nécessaire d'être dans le répertoire lib.

```
gcc -c bonjour.c -o bonjour.o
gcc -c bonsoir.c -o bonsoir.o
gcc -c bonnenuit.c -o bonnenuit.o
```

Question 5: Arbre de dépendance

Conclusion:
Le 1er makefile présent dans lib permet l'éxécution à la fois des programme dans lib et dans src.
Il faut cependant etre dans le répertoire lib pour utiliser la commande make.
Pour le second makefile, il faut être dans le répertoire salut-0.1 et les commandes, make, make install, ./salut éxécute le programme directement.

Exercice 2

Question 1:

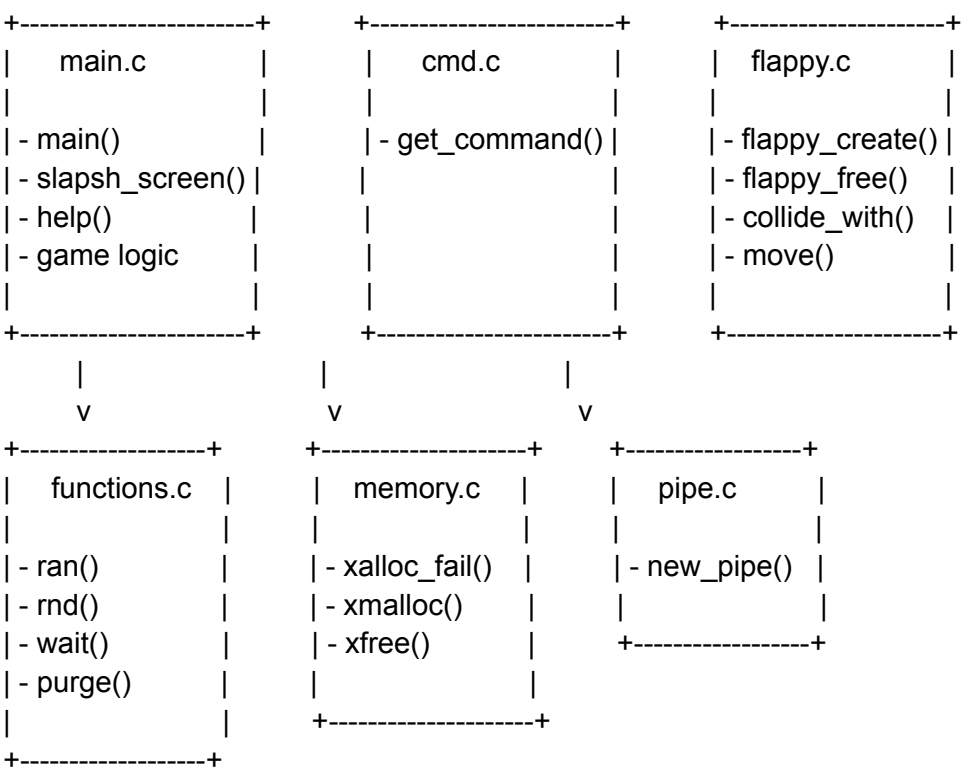
On peut factoriser les expressions précédentes, on obtiendrait donc :
CC : gcc
CFLAGS : -O3 -Wall -funroll-loops
LDFLAGS : -s
LIBPNG : -lpng

On peut donc créer une règle pour make:

```
pbm2png: pbm2png.c
	$(CC) $(CFLAGS_PBM2PNG) -o $@ $< $(LIBPNG)

pbm2eps9: pbm2eps9.o printer.o
	$(CC) $(LDFLAGS) -o $@ $^

pbm2eps9.o: pbm2eps9.c
	$(CC) $(CFLAGS_PBM2EPS9) -c -o $@ $<

printer.o: printer.c
	$(CC) $(CFLAGS_PBM2EPS9) -c -o $@ $<
```

Exercice 3

Question 2:



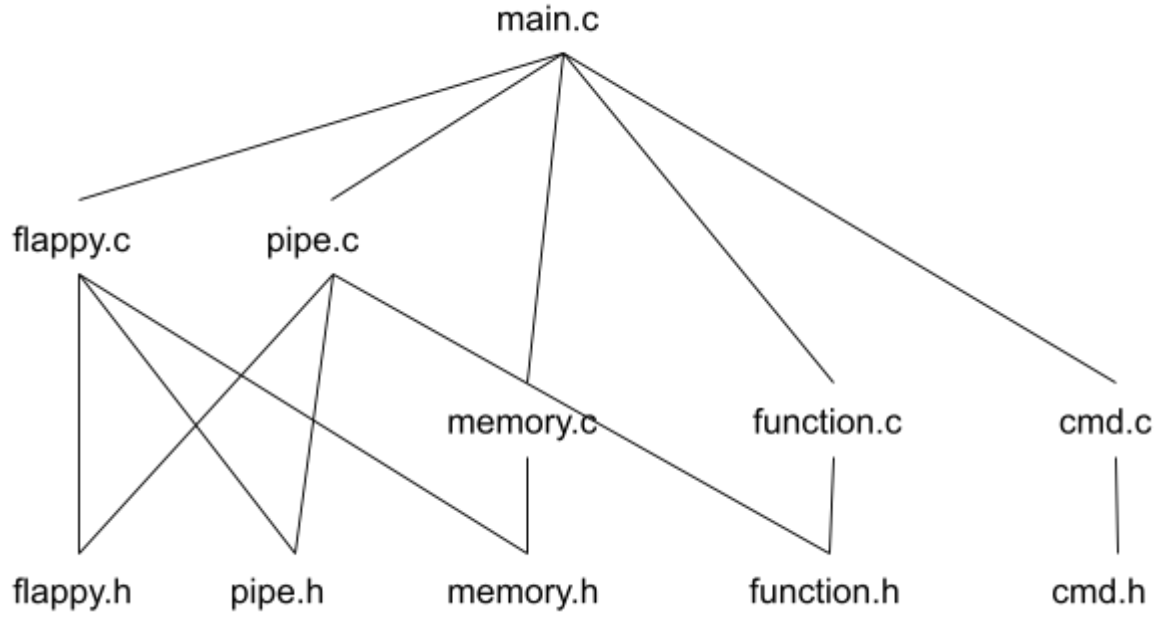
Dans cette représentation :

- Chaque rectangle représente un module du jeu.
- Les fonctions exposées par chaque module sont listées à l'intérieur du rectangle.
- Les flèches indiquent les dépendances entre les modules. Par exemple, main.c dépend de cmd.c, flappy.c, functions.c, memory.c, et pipe.c.
- Les fonctions du module flappy.c telles que flappy_create(), flappy_free(), collide_with(), et move() sont listées pour représenter son interface.

Question 3:

Pour obtenir un exécutable à partir des fichiers source C du jeu Flappy , on doit compiler tous les fichiers source nécessaires.
Tout en s'assurant que les fichiers header correspondants (.h) sont inclus dans la compilation pour chaque fichier source.
Arbre des dépendances du jeu FlappyTxt:

main.c:
  flappy.h
  pipe.h
  cmd.h
  functions.h
cmd.c:
  cmd.h
flappy.c:
  flappy.h
  memory.h
  pipe.h
  functions.h
functions.c:
  functions.h
memory.c:
  memory.h
pipe.c:
  pipe.h
  functions.h
  flappy.h
Voici l'arbre de dépendance schématisé.



Question 5/6

Voici les commandes à` utiliser pour obtenir le jeu en respectant les principes de la programmation modulaire tout en prenant en compte la syntaxe du standard C99:

gcc -std=c99 -Wall -Wextra -c -o main.o main.c
gcc -std=c99 -Wall -Wextra -c -o flappy.o flappy.c
gcc -std=c99 -Wall -Wextra -c -o cmd.o cmd.c

```
gcc -std=c99 -Wall -Wextra -c -o functions.o functions.c
gcc -std=c99 -Wall -Wextra -c -o memory.o memory.c
gcc -std=c99 -Wall -Wextra -c -o pipe.o pipe.c
gcc -std=c99 -Wall -Wextra -o flappy_game main.o flappy.o cmd.o functions.o memory.o
pipe.o
```