L'entrypoint est différent pour une librairie de jeu et graphique, afin de pouvoir différencier le type de librairie
“createGame” est l’entrypoint pour une librairie de jeu
“createGraph” est l’entrypoint pour une librairie de graphique

L'entrypoint renvoie un pointeur de la librairie en question

Class interface de librairie graphique

```
namespace Arc {
    class IDisplayModule {
        public :
            virtual ~IDisplayModule() = default;
            virtual void init(Arc::InfoGame info) = 0;
            virtual void display(Arc::InfoGame info) = 0;
            virtual Arc::Keys getKeyPress() = 0;
            virtual void stop() = 0;
            virtual const std :: string &getName() const = 0;
    };
}
```

IDisplayModule:
init: fonction qui prend en paramètre les informations du jeu pour l'initialisation de tout ce dont tu as besoin pour ta librairie graphique (par exemple : texture, taille de fenêtre...)
display: fonction qui prend en paramètre les informations du jeu pour les afficher
getKeypress: fonction qui transmet les inputs du clavier
stop: fonction pour fermer la fenêtre et tout ce qui est utilisé par la librairie graphique
getName: fonction qui retourne le nom de la librairie graphique

Class interface de librairie de jeu

```
namespace Arc {
    class IGameModule {
        public :
            virtual ~IGameModule() = default;
            virtual Arc::InfoGame loop_game(bool run) = 0;
            virtual int handleKeyPress(Arc::Keys c) = 0;
            virtual const std :: string &getName() const = 0;
            virtual void setHighestScore(std::size_t best_score) = 0;
    };
}
```

IGameModule:
loop_game: fonction prenant en paramètre un booléen indiquant si le jeu doit être en cours d'exécution. Cette fonction doit effectuer un tour de boucle du jeu si le booléen est "true" et retourner les informations du jeu
handleKeyPress: fonction prenant en paramètre l'input du clavier, retourne différentes valeurs en fonction de l'input du clavier en paramètre:

- Si l'input du clavier correspond à un changement de librairie graphique, retourner -1
- Si l'input du clavier correspond à un changement de librairie de jeu, retourner -2
- Si l'input du clavier correspond à un retour au menu, retourner -3
- Si l'input du clavier correspond à l'arrêt complet de l'arcade, retourner -4
- Si l'input du clavier correspond au fait de relancer le jeu actuel, retourner -5
- Si l'input du clavier correspond au fait de choisir une librairie dans le menu, retourner un nombre positif correspondant à l'index plus un de la librairie sélectionnée.
- Sinon retourner 0

getName: fonction qui retourne le nom de la librairie de jeu
setHighestScore: fonction prenant en paramètre une valeur spécifiant le score le plus élevé jamais obtenu dans le jeu

Classe pour connaître les différentes variantes d'un caractère.

```
namespace Arc {
    class Share {
        public :
            Share(char valc,  std::pair<std::size_t, std::size_t> valfrontclr, std::string path,
            std::pair<std::size_t, std::size_t> size,
            std::pair<std::size_t, std::pair<std::size_t, std::size_t>> color_rgb) :
            c(valc), ncurseclr(valfrontclr), image_path(path), image_size(size), rgb(color_rgb) {};
            ~Share() = default;
            char c;
            std::pair<std::size_t, std::size_t> ncurseclr;
            std::string image_path;
            std::pair<std::size_t, std::size_t> image_size;
            std::pair<std::size_t, std::pair<std::size_t, std::size_t>> rgb;
    };
```

Share:
c: un caractère de la carte du jeu sert de référence pour savoir quel variant de ce caractère on utilise en fonction des librairies graphiques
ncurseclr: la couleur en ncurse (frontcolor, backcolor)
image_path: le chemin de l'image
image_size: la taille de l'image ou de l'élément présent en cas d'absence de l'image (longueur, largeur)
rgb: couleur de l'élément en cas d'absence d'image

Class pour stocker un élément du jeu avec sa position
(se trouve aussi dans le namespace Arc)

```cpp
template<typename T> class GameStat {
    public :
        GameStat() {};
        ~GameStat() = default;
        T val;
        std::pair<std::size_t, std::size_t> pos;
};
```

GameStat:
val: valeur de l'élément
pos: position de l'élément

Class pour partager les informations entre la librairie graphique et la librairie de jeu
(se trouve aussi dans le namespace Arc)

```cpp
    class InfoGame {
        public :
            InfoGame() {};
            InfoGame(std::vector<Arc::Share> dico, GameStat<std::vector<std::string>> valmap,
            std::pair<std::size_t, std::size_t> pplayer, GameStat<std::size_t> valscore,
            GameStat<std::size_t> valhighest_score, std::pair<std::size_t, std::size_t> size_win,
            std::string text, GameStat<std::size_t> valtimer):
            assets_dico(dico), map(valmap), pos_player(pplayer), score(valscore),
            highest_score(valhighest_score), window_size(size_win),
            font(text), timer(valtimer) {};
            ~InfoGame() = default;
            std::vector<Arc::Share> assets_dico;
            GameStat<std::vector<std::string>> map;
            std::pair<std::size_t, std::size_t> pos_player;
            GameStat<std::size_t> score;
            GameStat<std::size_t> highest_score;
            std::pair<std::size_t, std::size_t> window_size;
            std::string font;
            GameStat<std::size_t> timer;
        };
```

InfoGame:
assets_dico:un dictionnaire qui nous permet d'utiliser un caractère pour connaître les correspondances de ce caractère à utiliser en fonction des librairies graphiques.
exemple: si on est dans une lib comme la SFML on utilise le path de l'image qui est lié à ce caractère
map: la carte du jeu avec sa position de début (le haut à gauche de la map)
pos_player: la position du joueur (x, y)
score: le score actuel du jeu avec sa position (x, y)
highest_score: le score le plus haut jamais obtenu de ce jeu avec sa position (x, y)
window_size: la taille de la fenêtre (x, y)
font: chemin pour accéder à la fonte du jeu
timer: le timer avec sa position (x, y)

Enum pour les touches du clavier

```
namespace Arc {
    enum Keys {
        NONE,
        A, B, C, D, E, F, G, H, I, J, K, L, M,
        N, O, P, Q, R, S, T, U, V, W, X, Y, Z,
        LEFT, RIGHT, UP, DOWN,
        ENTER, SPACE, DELETE, BACKSPACE, TAB, ESC
    };
};
```