Façade du bâtiment

Commentaire de spécification

Procedure1 : seDeplacer

Procedure2 : cadreFenetre

Procedure3 : fenetreDuBas

Procedure4 : fenetreDuHaut

Procedure5 : case

Procedure : programme principal

Procedure1 : seDeplacer

Objectifs : Pouvoir se déplacer dans le repère

Méthode : utilisation des fonctions up(), goto() et down() de turtle

Besoins : x, y

Entrée : x, y

Sortie : -

Connu : -

Résultat :

Hypothèse :

Procedure2 : cadreFenetre

Objectifs : définir le grand rectangle qui contient toutes les fenêtres

Méthode : utilisation des fonctions seDeplacer() et rectangle()

Besoins : a, b

Entrée : a,b

Sortie : -

Connu : -

Résultat : -

Hypothèse : -

Procédure3 :fenetreDuBas

Objectifs : créer les fenêtres du bas

Méthode : utilisation des fonctions seDeplacer(), rectangle() et goto()

Besoins : x, y

Entrée : x, y

Sortie : -

Connu : -

Résultat : -

Hypothèse : -

Procédure4 : fenetreDuHaut

Objectifs : créer les fenêtres du haut

Méthode : utilisation des fonctions seDeplacer(), carre() et goto()

Besoins : x, y

Entrée : x, y

Sortie : -

Connu : -

Résultat : -

Hypothèse : -

Procédure5 : Case

Objectifs : créer les cases qui sont en haut du bâtiment

Méthode : utilisation de la boucle, de la fonction rectangle() et seDeplacer()

Besoins : x, y

Entrée : x, y

Sortie : -

Connu : -

Résultat : -

Hypothèse : -

Procédure : Programme principal

Objectifs : appeler toutes les fonctions pour construire la façade du bâtiment

Méthode : utilisation des fonctions rectangle(), carre(), dot(), trapeze(), demi_cercle(),

Left(), right(), seDeplacer(), cadreFenetre(), fenetreDuHaut(), fenetreDuBas(),case()

Besoins :

Entrée :

Sortie : -

Connu : -

Résultat : -

Hypothèse : -

Diagramme des flux

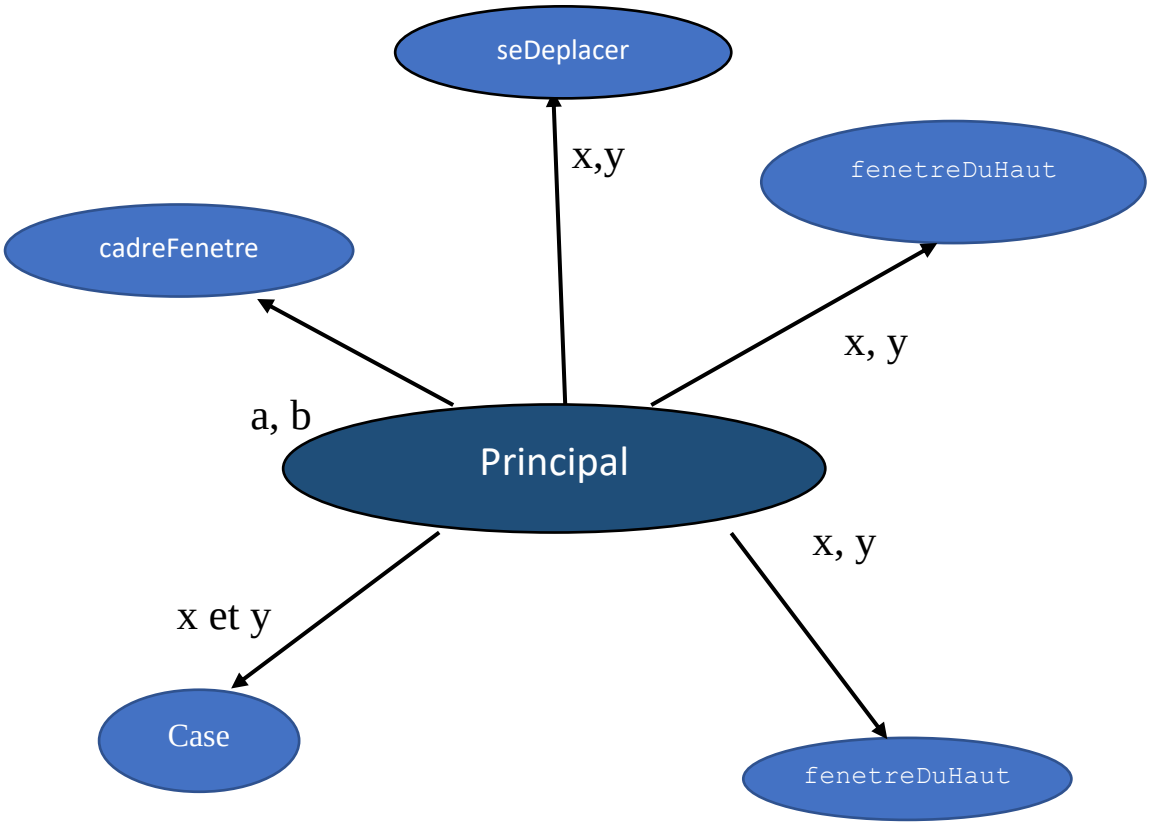


Tableau des flux :

PROGRAMME PRINCIPAL	FOURNIT (ENTREES)	REÇOIT (SORTIES)
sedeplacer	x, y	-
cadrefenetre	a, b	-
fenetreDuHaut	x, y	-
fenetreDuBas	x, y	-
case	x, y	-

MAQUETTE D'UN PONT

Procédure : Programme principale

Procédure 1 : arc

Procédure 2 : tracer

Procédure 3 **:** tracerPuisRetour

Procédure 4 : grandTriangle

Procédure 5: seDeplacerSansTracer

Procédure 6 : piedPont

Commentaires de spécification :

- Procédure 1 : arc
  - Objectif : dessiner un arc en fonction de **rayon** et **angle**

- Méthode : utilisation des fonctions circle( ) et position( ) de turtle
- Besoins : rayon, angle, debut
- Connus : -
- Entrées : rayon, angle, debut
- Sorties : -
- Résultats : obtenir un arc
- Hypothèses : rayon > 10 et angle > 0

- Procédure 2 : tracer
  - Objectif : tracer un segment en fonction de la distance
  - Méthode : utilisation de la fonction forward( ) de turtle
  - Besoins : distance
  - Connus : -
  - Entrées : distance
  - Sorties : -
  - Résultats : obtenir un segment de longueur **distance**
  - Hypothèses : distance != 0

- Procédure 3 : tracerPuisRetour
  - Objectif : tracer un segment en fonction de la **distance** avec une direction suivant la valeur de la variable **angle** puis retourner au point de départ
  - Méthode : utilisation des fonctions setheading( ), forward( ) et back( ) de turtle
  - Besoins : angle, distance
  - Connus : -
  - Entrées : angle, distance
  - Sorties : -
  - Résultats : obtenir un segment de longueur **distance** suivant la direction en fonction de **angle**
  - Hypothèses : distance != 0

- Procédure 4 : grandTriangle
  - Objectif : tracer un triangle en fonction de cote1 et cote2
  - Méthode : tracer le triangle rectangle gauche puis le triangle rectangle droite en utiliant les fonctions forward( ), setheading( ) et goto( ) de turtle
  - Besoins : cote1, cote2
  - Connus : -
  - Entrées : cote1, cote2
  - Sorties : -
  - Résultats : obtenir un triangle
  - Hypothèses : cote1 != 0 et cote2 != 0

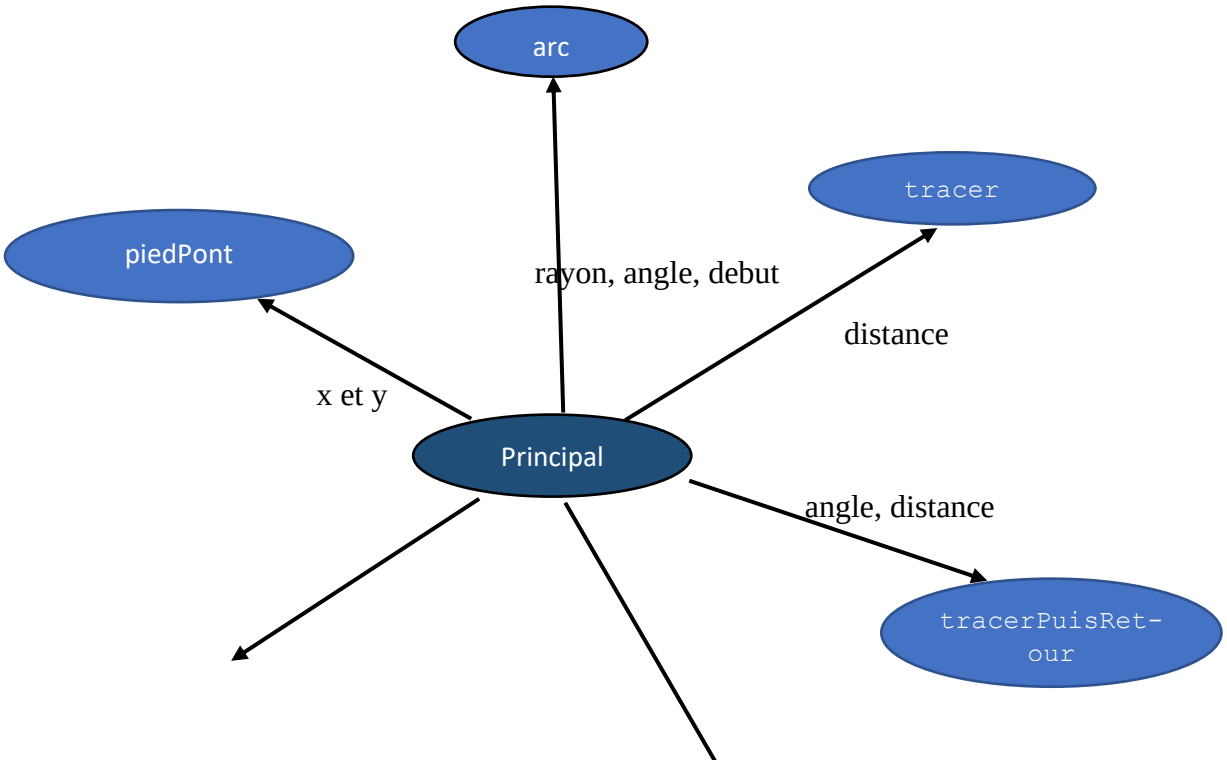
- Procédure 5 : seDeplacerSansTracer
  - Objectif : se déplacer en fonction des coordonnées x et y
  - Méthode : utilisation des fonctions position( ) et goto( ) de turtle
  - Besoins : x et y
  - Connus : -
  - Entrées : x et y
  - Sorties : -
  - Résultats : déplacement vers le point de coordonnées x et y

- Hypothèses : -

- Procédure 6 : piedPont
  - Objectif : tracer un rectangle plein en fonction de x et y
  - Méthode : utilisation de la fonction rectangle de notre bibliothèque `genieCivilOuvrage2D` et la fonction de remplissage de `turtle`
  - Besoins : x et y
  - Connus : -
  - Entrées : x et y
  - Sorties : -
  - Résultats : obtenir un rectangle plein
  - Hypothèses : x != 0 et y != 0

- Procédure : programme principale
  - Objectif : faire appel aux fonctions pour obtenir une maquette de pont complète
  - Méthode : appel de fonction et usage de paramètres formels et effectifs, avec l'affectation de valeurs directement pour ces paramètres
  - Besoins : rayon, angle, debut, distance, cote1, cote2, x, y
  - Connus : -
  - Entrées : rayon, angle, debut, distance, cote1, cote2, x, y
  - Sorties : -
  - Résultats : obtenir une maquette de pont complète
  - Hypothèses : rayon > 10, angle > 0, distance > 0, cote1 > 0, cote2 > 0, x != 0, y != 0

DIAGRAMME DES FLUX :



x et y

cote1, cote2

TABLEAU DES FLUX :

Programme Principal	Fournit (Entrées)	Reçoit (Sorties)
arc	rayon, angle, debut	-
tracer	distance	-
tracerPuisRetour	angle, distance	-
grandTriangle	cote1, cote2	-
seDeplacerSansTracer	x, y	-
piedPont	x, y	-