Project 2 Flowchart

Author: Zachary Lyman
Created on February 12 , 2022, 4:00 PM
Purpose: Play Roulette

System Libraries
I/O Objects
Formatting Library
String Library
C Standard Library
Time Library
File I/O
Math Library
Vector Library

User Libraries
none

Global Constants
none

Function Prototypes
int spinWhl
int asgnNum
string asgnClr
void dpslMnu
bool choice
void readlog
sgleNum
unsigned char readSel
float readBet
void dspBL
void dspSL
void bublSrt
void fillAns
void bblSrtA
int binSrch
int gameCnt
void dspGCnt

main

Output to user
Game header
How much the the user has
Prompt to user menu
Min bet reminder

Initialize Random Seed
srand(static_cast<unsigned int>(time(0)))

Declare Variables
ROWS
COLS
ANSWR
random
space
indx
sglNum
gCount
games
wheel
ans
betNum
select
bet
betLog
selLog
usrBank
betAmnt
sel
betSel
ballPos
logLine
color
ballLog
quit

Initialize Values
usrBank=200
sel='B'
gCount=0

Write to file
File:BallLog.txt
Output:"Ball Log"

void fillAns
(ans,ANSWR)

void bblSrtA
(ans,ANSWR)

Page 2

void fillAns(int a[],int n)

Out of Loop
i<n

In For Loop
i=0

i<n

False

True

i++

a[i]=rand()%90+10

main

page 2

page 2

d-wloop
select!='q' or 'Q' ← D

Output game header

bool choice
(select,bet,
usrBank,betNum)

bool choice==true —False→

True ↓

sel='q'

indx=0

random=rand()%720

spinWhl
(wheel,COLS,
random)

indx=int spinWhl

asgnNum
(wheel,Cols,
indx)

space=int asgnNum

asgnClr
(wheel,Cols,
indx) → color=string asgnClr → page 3

---

void bblSrtA
(a[],int n)

Declare
swap

d-wloop
swap==true

swap=false
n--

Out of Loop
i<n

In For Loop
i=0

i++

a[i]>a[i+1] —True→ swap=true
int temp=a[i]
a[i]=a[i+1]
a[i+1]=temp

False ↓

swap==true —False→ main

True

---

int spinWhl
(int array[][38],int
size,int random)

float indx
indx=random%38

return floor(indx)

---

int asgnNum
(int array[][38],int size
int random)

int space
space=array[0]
[indx]

return space

---

string asgnClr
(int array[][38],int size
int random) → int colNum
string color
colNum=array[1]
[indx]

colNum==1 —True→ color="black"

False ↓

colNum==2 —True→ color="red"

False ↓

color="green" → return space

Left flowchart

page 3 (circle)

↓

Out of Loop
i<select.size() ← **B** (circle)

↓

In For Loop
i=0

↓

i<select.size() (decision) → **C** (circle)
False

True ↓

betAmnt=0
betAmnt=bet[i]
usrBank=usrBank-
betAmnt
betSel=select[i]
sglNum=betNum[i]

↓

i==0 (decision) → False → **A** (circle)

True ↓

space>0
& <37 (decision) → True → **Output to file**
space
color (parallelogram) → **A**

False ↓

space==37 (decision) → False → **ballPos="00"** → **Output to file ballPos**

True ↓

ballPos="0" → **Output to file**
ballPos (parallelogram) → **A**

Right flowchart

void bblSrt
(vector<unsigned
char>&selLog,
vector<float>&betLog) (ellipse)

↓

unsigned int
n=selLog.size()
swap

↓

d-wloop
swap==true

↓

swap=false
n--

↓

Out of Loop
i<n

↓

In For Loop
i=0

↓

selLog[i]>
selLog[i+1] (decision) → True → **swap=true**
int tempCH=selLog[i]
selLog[i]=selLog[i+1]
selLog[i+1]=tempCh
tempFl=betLog[i]
betLog[i]=betLog[i+1]
betLog[i+1]=tempFl → **i++**

False ↓

swap==true (decision) → False → **main** (ellipse)

True

A

random<37 — True → bet result header → color==black — True → color → betSel=='b' or 'B' — False → betSel=='r' or 'R' — False

betSel=='b' or 'B' — True → usrBank+=betAmnt*2 → Display Win

betSel=='r' or 'R' — True → Display Loss

random<37 — False

color==black — False → color==red

Display Win → push win/loss to betLog

color==red — True → color → betSel=='r' or 'R' — False → betSel=='r' or 'R' — False

betSel=='r' or 'R' — True → usrBank+=betAmnt*2 → Display Win → push win/loss to betLog

betSel=='r' or 'R' — True → Display Loss

color==red — False → space%2==0 — True → betSel=='e' or 'E' — False → betSel=='o' or 'O' — False

betSel=='e' or 'E' — True → usrBank+=betAmnt*2 → Display Win

betSel=='o' or 'O' — True → Display Loss

Display Win → push win/loss to betLog

space%2==0 — False → space%2==1 — True → betSel=='o' or 'O' — False → betSel=='e' or 'E' — False

betSel=='o' or 'O' — True → usrBank+=betAmnt*2 → Display Win

betSel=='e' or 'E' — True → Display Loss

push win/loss to betLog

space%2==1 — False → space >=1 and <19 — True → betSel=='l' or 'L' — False → betSel=='h' or 'H' — False

betSel=='l' or 'L' — True → usrBank+=betAmnt*2 → Display Win

betSel=='h' or 'H' — True → Display Loss

push win/loss to betLog

space >=1 and <19 — False → space >19 and <37 — True → betSel=='o' or 'O' — False → betSel=='e' or 'E' — False

betSel=='o' or 'O' — True → usrBank+=betAmnt*2 → Display Win

betSel=='e' or 'E' — True → Display Loss

Display Win → push win/loss to betLog → push betSel to selLog → page 5

space >19 and <37 — False → page 5

space ==37 — False → ballPos="00" → Display Results

space ==37 — True → ballPos="0" → Display Results

select =='s' or 'S' — True → sglNum ==space — True → usrBank+= (35*betAmnt)+betAmnt → push win/loss to betLog → page 5

select =='s' or 'S' — False → Display Loss

sglNum ==space — False → Display Loss

Page 5 (flowchart)

(Page 5) → Display usrBank to user

- Decision: usrBank<5
  - True → Display game over → sel='q'
  - False →
- Decision: select=='q' or 'Q'
  - True → Display Payout → i++
  - False → i++
- i++ → **(B)**

---

int gameCnt(int gCount)
→ static int count=1
→ gCount=gCount+count
→ return gCount

---

void dspGCnt (int games, int lghtSp)
→ display games played games-1
→ display message reaches at lghtSp

(C) → clear select clear bet clear betNum
→ int gameCnt(gCount)
→ games=games+gameCnt(gCount)
- Decision: d-whloop sel!='q' & 'Q'
  - True → **(D)**
  - False →
→ void bublSrt (selLog,betLog)
→ void dspSL (selLog)
→ void dspBL (betLog)
→ display payout game footer
→ display ultimate question
→ val=42
→ int binSrch (ans,ANSWR,val)
→ ansI=binSrch
- Decision: ansI != -1
  - True → display ultimate question answer → dspGCnt(games)
  - False → display ultimate question answer not found → dspGCnt(games)
→ dspGCnt(games) → return 0

int binSrch (int a[],int n, int val)

int high=n-1, low=0

d-whloop low<high

int middle= (high+low)/2

val = a[middle] — True → return middle

False

val < a[middle] — True → high=middle-1

False

low=middle-1

low<high — True

False

return -1

void dspBL (vector<float>&betLog

i<betLog.size() — False → main

True

Out for loop i<betLog.size()

In for loop i=0

Display betLog[i]

i==5 — False → i++

True

Display endl

void dspSL (vector<float>&betLog

i<selLog.size() — False → main

True

Out for loop i<selLog.size()

In for loop i=0

Display selLog[i]

i==5 — False → i++

True

Display endl

void log (fstream &ballLog, string logLine)

Display logLine from file

bool choice

```
bool choice
(vector<unsigned int>&select,
vector<float>&bet,
float usrBank,
vector<int>&betNum
```

```
repeat
s
b
ballLog
logLine
```

```
repeat!='n'
'N'
```
True → G

False → return false

```
d-wloop
repeat!='n'
'N'
```

dpslMnu(usrBank)

repeat='n'

```
void dsplMnu
(flaot usrBank)
```

Display menu items usrBank

Prompt for input s

```
s=='x'
'X'
```
True → exit (0)

False

G →
```
repeat=='y'
'Y'
```
True →
```
d-wloop
s=='q'
'Q'
```
→ Prompt for new input →
```
s=='q'
'Q'
```
True / False

False

```
s!='q'
'Q'
```
True →
```
s=='p'
'P'
```
True → log (ballLog,logLine) → Prompt for new input → push s to select

False → return true

False →
```
s=='s'
'S'
```
False → push 1 to betNum → E

True → int num → Prompt for single number bet → push num to betNum → Prompt bet dollar input

E → Prompt bet dollar input

```
d-wloop
b<5
```

```
b<5
```
False →
```
d-wloop
b>usrBank
```
→
```
b>usrBank
```
True → Prompt for new dollar bet

False → push num to betNum → usrBank-b → Prompt for more bets

True → Prompt for new dollar bet

```
ustBank<5
```
False → F

True → repeat='n' → F