Cigarette smokers problem

The cigarette smokers problem was originally presented by Suhas Patil. Four threads are involved: an agent and three smokers. The smokers loop forever, first waiting for ingredients, then making and smoking cigarettes. The ingredients are tobacco, paper, and matches.

We assume that the agent has an infinite supply of all three ingredients, and each smoker has an infinite supply of one of the ingredients; that is, one smoker has matches, another has paper, and the third has tobacco.

The agent repeatedly chooses two different ingredients at random and makes them available to the smokers. Depending on which ingredients are chosen, the smoker with the complementary ingredient should pick up both resources and proceed.

For example, if the agent puts out tobacco and paper, the smoker with the matches should pick up both ingredients, make a cigarette, and then signal the agent.

My solution uses mutex, conditional variables and shared memory. See canvas for the entire solution, here I just show the code the threads call.

```
int whoseTurn(){
  int turn;

  if ((tobacco == 0) && (matches == 0) && (paper == 0)){
    turn = 0;  //agent turn
  }
  if ((tobacco == 0) && (matches == 1) && (paper == 1)){
    turn = 1; //tobacco turn
  }
  if ((tobacco == 1) && (matches == 0) && (paper == 1)){
    turn = 2; //matches turn
  }
  if ((tobacco == 1) && (matches == 1) && (paper == 0)){
    turn = 3; //paper turn
  }

  return turn;
}

void canThreadContinue(int id){
  int turn;   //remember, 1 = tobacco, 2= matches, 3 = paper , save 0 for agent
  pthread_mutex_lock(&TABLE);
  turn = whoseTurn();
  while( turn != id){
    switch(id){
      case 1:
        printf("smoker %d (that is tobacco) darn, the goods on the table are not for me, i have to wait\n",id);
        break;
      case 2:
        printf("smoker %d (that is matches) darn, the goods on the table are not for me, i have to wait\n",id);
        break;
      case 3:
        printf("smoker %d (that is paper) darn, the goods on the table are not for me, i have to wait\n",id);
        break;
    }
    pthread_cond_wait(&self, &TABLE);
    turn = whoseTurn();
  }
  pthread_mutex_unlock(&TABLE);
  return;
}

void smoker(int * id){

  while(1){
    sleep(1);  //getting ready
    canThreadContinue(*id); //remeber, this will BLOCK the thread IF the condition is not met

    switch(*id){
      case 1:
        printf("smoker %d (that is tobacco) Yippe it is for me, time to smoke\n",*id);
        break;
      case 2:
        printf("smoker %d (that is matches) Yippe it is for me, time to smoke\n",*id);
        break;
      case 3:
        printf("smoker %d (that is paper) Yippe it is for me, time to smoke\n",*id);
        break;
    }
    tobacco = 0; matches = 0; paper= 0;
    pthread_cond_broadcast(&self);//wake all threads wating on the conditional variable
    sleep(1);  //do some smoking
  }
}
void agent(int * times){
  int rn,x=0;
  while(x < *times){
    canThreadContinue(0);  //remeber, this will BLOCK the thread IF the condition is not met
    rn=rand()%3 + 1; //1=tobacco, 2=matches, 3=paper -- reserve 0 for the agent,
    switch (rn){//set the global variables to be needs of a smoker
      case 1:
        printf("agent placed matches and paper on table\n");
        tobacco = 0; matches = 1; paper= 1;
        break;
      case 2:
        printf("agent placed tobacco and paper on table\n");
        tobacco = 1; matches = 0; paper= 1;
        break;
      case 3:
        printf("agent placed matches and tobacco on table\n");
        tobacco = 1; matches = 1; paper= 0;
        break;
    }
    sleep(1);  //putting material on table
    pthread_cond_broadcast(&self);//wake all threads wating on the conditional variable
    x++;
  }
  printf("agent done\n");
  pthread_exit(NULL);
}
```